Spring 2025

Building blocks to boost your DevOps transformation

Adaptavist

Building blocks to boost your DevOps transformation

Contents

Introduction

DevOps has gone mainstream. According to HGS Digital's A Closer Look at DevOps, only 3 percent of software development businesses surveyed said they had never heard of DevOps in 2018, and just 9 percent said they had no plans to adopt it into their workflow. More recently, adoption has continued to soar, with 80 percent of companies now using DevOps practices (compared to just 33 percent in 2017). So much so, the DevOps market is projected to expand from US$13.29 billion in 2024 to an impressive US$108.26 billion by 2035.

Rather than asking themselves whether DevOps is something they should prioritise, senior leaders are already onboard. Now, they're focused on ensuring that they're using DevOps to their advantage. But with so many organisations confidently jumping on the bandwagon, many claiming to be dedicated converts, what does DevOps really look like on the ground? And how intrinsically has the framework been adopted?

What does this mean to do DevOps?

‘We have an experienced DevOps team’ or ‘We’re hiring a DevOps engineer’ are the types of sentiments commonly bandied around, but they’re big red flags. DevOps isn’t a tool or individual process restricted to a few people or isolated teams; it’s a mindset shift, much like agile, and applies to IT as a whole. It shouldn’t be a standalone department but rather a continuous culture that takes root and filters throughout the business.

The adoption of DevOps changes a business's culture, allowing teams to work fast and with autonomy. As author and entrepreneur Gene Kim said: ‘[DevOps is] really about enacting technology practices and architecture that help organisations create a very fast – or even continuous – product flow, from dev through testing through operations, all the way to deployment, while often increasing reliability, stability, and security.'

This is directly opposed to more traditional models. In the past, a software company would have to involve hundreds of developers over a period of weeks to deploy small upgrades or fix bugs. A waterfall model – a linear approach where changes are batched up – made releases unwieldy and risky, not to mention the time it would take to release a new product. DevOps developed as a direct response to these inefficiencies: slow turnaround, poor quality, lack of accountability and poor teamwork in particular. Today, high-level DevOps adopters can release reliable products at a much faster pace.

In recent years, the practice and definition of DevOps has expanded. It now includes GitOps – whereby Git repositories are leveraged as the single source of truth and infrastructure as code (IaC) is employed to enhance traceability, simplify rollbacks, and foster collaboration.

Platform engineering is another extension to traditional DevOps. The use of self-service platforms that abstract infrastructure and provide standardised tools and workflows empowers developers to make logical, streamlined decisions, rather than get caught up in operational complexity.

DevOps is going beyond agile

Agile is a well-established approach – it celebrated its 20th birthday in 2021. By now, most organisations have made or are in the process of some kind of agile transformation. But however hard you're waving your agility wand to accelerate product development, drive value, and overhaul your organisation, it’s really all for nothing if you can’t get your products to market at the same rate. No matter what comes before it, if your software is still worming its way through traditional sign-offs and endless testing, building up to a big release, then you're not doing agile properly – and you’re certainly not doing DevOps.

DevOps is agile's spunky younger sister. Where agile is more focused on development, DevOps works to accelerate delivery, making it faster while maintaining quality and ensuring continuous development with tight feedback loops. It helps you carve a path from ideation to production, uniting software development and tech operations for an end-to-end pipeline that includes every stakeholder in the process.

As a combination of organizational philosophies, values, principles, tools, and practices, DevOps requires people to change the way they work and the way they think across all aspects of development and operations – from front-end developers to back-end IT infrastructure and security professionals.

Bringing it all together

Where agile emphasises adaptive planning, collaboration, and incremental delivery, DevOps automates processes and fosters cross-functional collaboration. In practice, iterative sprints associated with agile are complemented by DevOps' continuous integration and delivery (CI/CD) pipelines. That means agile teams can develop features incrementally while DevOps practices ensure any updates are built, tested, and deployed automatically.

This synergetic approach helps to reduce hand-off delays, enhance feedback loops, and minimise time to market for new features. And customers' needs remain at the centre of it all, with value and innovation being delivered more frequently in a reliable and scalable way.

The three tenets of DevOps

Three fundamental principles underpin DevOps:

Flow	**Feedback**	**Continuous learning and experimentation**

Together they ensure teams are united towards a common goal and reaping the rewards of continuous integration (CI) and delivery (CD). Its success rests on three core components:

People	**Processes**	**Tools**

Over the following chapters, we'll take a look at each in turn, providing an overview of the key building blocks you need to consider and providing our top tips to help you step up your DevOps practices.

Part one:

Empowering people, changing culture

Looking back, old-school software development seems a little silly. With its siloed developers and system architects waving over a metaphorical wall at the operations engineers and database administrators on the other side, it's questionable that these two fundamental facets of getting software out the door haven't always been on the same page.

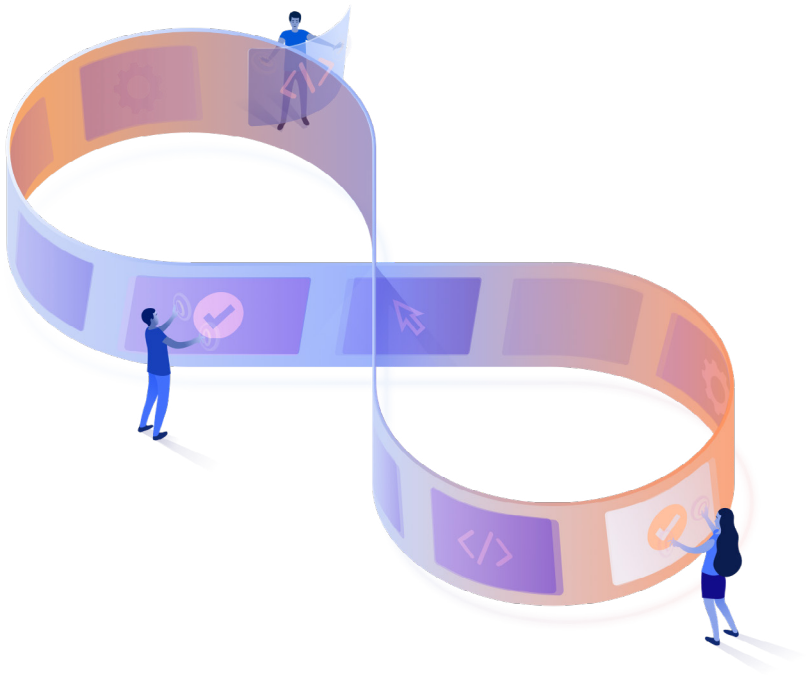
While it might sound like a logical solution to overcoming process inefficiencies, for some people, transitioning to DevOps is a big ask. And it's way out of their comfort zone. This is particularly true of operations teams, who are used to working to reduce risk by minimising change and taking things slow. Now they're being expected to speed up with continuous integration (CI) and continuous delivery (CD), release regularly, and operate in a fail-fast pipeline. DevOps aims to bridge the gap between developers' change-ready ideology and operations' more cautious approach.

For DevOps to be successful, everyone has to be on board. If your organisation understands and supports that a compatible culture will develop and the new hires, tools, and tech you need to make it happen will emerge in its wake. In this chapter, we'll explore what a DevOps culture looks like, how to get yours on track, and the people and skills you need to help it flourish.

A conducive culture

No one said DevOps was easy. In fact, Gartner once predicted that through 2022, 75% of DevOps initiatives would fail to meet expectations. The reason won't surprise you – issues around organisational learning and change. Simply showing up one day and imposing a new set of practices on your employees is dooming DevOps to failure.

Employee buy-in is paramount – knowing why the business is expecting something new from them and how it's going to make their work better and easier will reduce resistance significantly. Think of it as a really helpful, not-at-all-weird cult. People might join a cult on a whim, but they're not going to stick around and proliferate its ideologies and rituals for no reason. They need to understand the deeper thinking and the benefits – both to them personally and the wider group – to get stuck in. Here are some key considerations to build buy-in and transform your culture.



Here are some key considerations to build buy-in and transform your culture.

- **Educate with feeling**

  Sharing information about how DevOps will make a real difference is the first step to getting people onboard. Implement a change management strategy that considers the relative understanding, resistance, and experience of all teams. Everyone needs to come along for the ride, but it will be bumpier for some than others.

  The main goal is to give people a clear idea of how it's going to feel to work differently. The only way they will know this is by doing. So share actionable insights which they can apply to their work and observe the results, no matter where they are in the pipeline. Take this a step further with a program that simulates the experience of working in a fully DevOps environment.

- **Do it gradually**

  No one likes a sudden shock to the system, and DevOps is no different. A gradual change where employees can merge into this new mindset will get better results in the long run. While driving change from the top is vital, actual change will start at the bottom. By implementing DevOps at the team level, it enables individuals to realise the possibilities, discover potential pitfalls, and overcome them – crucially before the whole business has made a shift. This slow-burn approach allows people to redefine their roles at a more realistic pace too.

- **Collaboration is key**

  Being on the same page means actually getting stuck in together. Without collaboration – from individual developers to C-suite leaders –DevOps just doesn't work. In larger corporate structures where people and thinking are siloed, this is a big challenge. Communication plays a huge part in this. Employees should feel empowered to share knowledge with each other, for the betterment of everyone. Consider using community forums or collaboration tools (see chapter three).

- **Banish blame and embrace trust**

  Forget the blame game; it won't help you there. It's vital that people feel they can share information about failure and frustration across the wider organisation without fear of the pointed finger. Start from a place of trust. In the past, people were so confident in control frameworks, checklists, and risk mitigation, that they didn't really need to trust their colleagues. But with DevOps trust is earned through collaboration and small wins (that quickly scale). Trust your teams to act responsibly, share their failures honestly, and to uphold the principles and requirements outlined by the organisation.

- **Good enough is really good**

  People need to know that perfection is not the goal and that improvement – in process, quality, and speed – is what they're striving for. 'Good enough' shouldn't be considered a dirty phrase. It's what allows people to put something out there quickly, see how it works, and then iterate any issues away based on real-world feedback. This is why fluidity and flexibility in DevOps are so important: systems should support a fail-fast way of working rather than stifle it.

- **Budget for the best scenario**

  You're going to need really good people to enable CI and CD pipelines to work effectively. Don't underestimate your needs: make sure experienced, senior people are involved from the outset, and invest in the right tools too. It's going to take time to achieve your goals, but you'll get there much faster if you budget accordingly rather than sidelining your requirements at the start.

- **Rewards based on reality**

  A thriving, motivated workplace culture is one where people feel rewarded and strive towards common goals. Rather than reward teams for following protocols and meeting targets – especially targets that might even conflict with those of other teams – DevOps uses strategic metrics to determine success. Throughout the organisation, objectives should be centred around business outcomes and measured accordingly.

- **Take the initiative**

  Lastly, there are a number of key culture change initiatives that allow organisations to take direct action to help their culture evolve in the ways already outlined. These include blameless post-mortems, which encourage honest discussions and prioritise understanding what went wrong and how to prevent recurrence over assigning blame.

  Or how about DevOps Dojos? These immersive, hands-on learning sessions bring cross-functional teams together to solve real-world challenges. Guided by experienced coaches, who mentor participants in DevOps principles and tools, they help your people to learn by doing and encourage experimentation. This helps accelerate your cultural transformation and drives adoption of DevOps practices.

The right stuff

If it's not clear by now, people are the most important component of your DevOps transformation. Making sure you have the right talent to ignite the spark and carry the torch is half the battle. Despite the tech-heavy nature of software development, when it comes to DevOps, soft skills – like communication, collaboration, flexibility, and decision-making – should be prioritised alongside security, coding, and infrastructure knowledge. Don't restrict your DevOps skills audit to IT teams alone: make sure this mix of skills is present across the organisation.

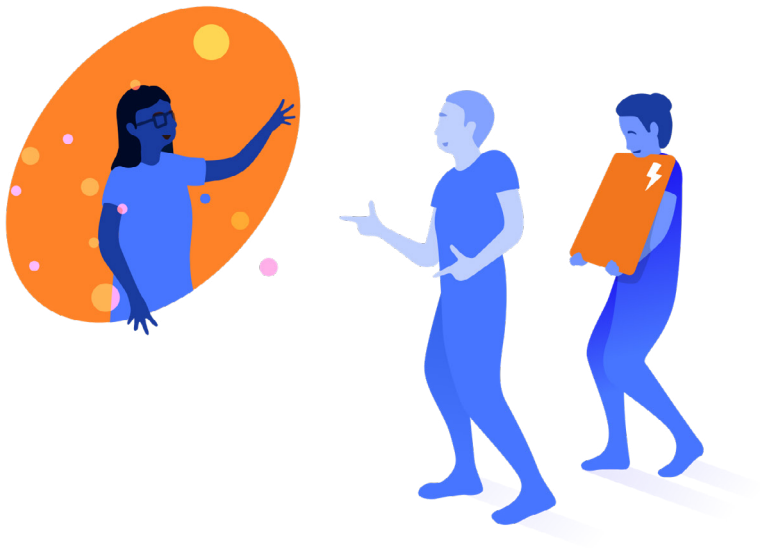
When hiring or upskilling your existing teams, don't forget the following:

- **Expertise takes time:** employees are not going to get up to speed in a week. There are new methodologies and tools to learn, so have patience. But don't be naive and expect people to pick things up on an ad hoc basis – self-learning during downtime won't cut it.
- **Program your people to win:** dedicate the right time and resources to helping your teams transition their skillset. One option is a DevOps training program that helps them become familiar with the tools, tech, and methods pivotal to transformation. FedEx took this approach and upskilled 25,000 programmers with its own bespoke training solution.
- **Know when to bring in new people:** it's important to strike the right balance between upskilling your existing employees and bringing in new people to boost your DevOps capabilities. Your existing employees might already have in-depth knowledge about your organisation, but there are lots of benefits to bringing new hires on

board. Keep in mind the cost of hiring in DevOps skills and that you'll need a certain level of expertise to pick the right people too.

- **Access your technology solutions:** make sure the tooling you're choosing is flexible enough to support the wider needs of the organisation. Everyone needs to get up to speed on DevOps, but only some people will develop an expert knowledge of tools. Others can simply benefit from automation and efficiency.

People play a huge part in your transformation, so don't sideline them or overlook their needs. And bring everyone along for the ride. Make sure teams are continuously informed, encouraged, and supported as they learn and acquire DevOps skills, and you'll reap the rewards.



Part two:

Making progress with process improvements

No matter the people in your teams or the tools you choose to use, successful DevOps transformation can't happen unless you step away from traditional IT's 'trust but verify' methodology to a new outlook of continuous integration and delivery, incorporating a set of essential processes. And a pick 'n' mix approach won't do here.

While organisations using DevOps will incorporate variations on the theme, they will all be utilising the following at a minimum: collaboration, automation, continuous integration and delivery, and monitoring. In addition, many organisations are now incorporating value stream management (VSM) and site reliability (SRE) practices too.

These fundamental processes will form the foundation of your DevOps pipeline, allowing everyone to work more efficiently and effectively towards achieving faster, high-quality software to suit the end users' needs.

The right stuff

Collaboration is more than just a cultural facet – it's a process-driven practice that you must build into the way you work; in particular, it helps IT organisations with globally distributed teams to concentrate on aligning to achieve a common goal, whether that's delivering a release by the end of the day or a wider business objective for the year. Here are just some of the problems collaboration can help you overcome:

- Operational inefficiencies, including limited automation and release delays
- Problematic handoffs caused by different time zones and disparate teams
- Redundant work and limited team problem-solving because of communication silos
- Impromptu changes that impact product stability
- Inbox overload and unnecessary meetings
- Uncertainty over roles and responsibilities

Top tips for collaboration

- **Use the right tools**
  Feedback and communication are key to ensuring collaboration is a constant and continuous process. Tools like Slack, Microsoft Teams, and Google Chat make it much easier to stay in touch and on top of what everyone's working on.

- **Set clearly defined goals**
  Make sure organisational, team-level, and individual goals are well-defined with clear objectives. Big-picture goals should be the same across the board so everyone's priorities are aligned.

- **Embrace diversity**
  Your team might be geographically spread out and culturally diverse, so inclusivity and sensitivity are key to smooth collaboration. Don't make assumptions, and embrace different points of view; it will help to unite the team in the long-run

- **Emphasise equality and unity**
  For everyone to feel invested, they need to know their contribution is valued. Unifying language—face–to–face and through digital communications—and leaders who inspire honesty and respect help create a sense of oneness and equality.

Automation: repetition right at your fingertips

What sets DevOps apart is automation. It's only possible to achieve this level of efficiency by relying heavily on automated processes to lighten the load. For that, you need to have the right suite of tools in place (see part three) for automation, building, integration, testing, environment configuration, deployments, workflows, data synchronisation across systems, and more, leaving people to focus less on repetitive tasks and more on complex problems.

If you already have some level of automation in place or are considering automating something new, consider how the following techniques can help improve your efforts and boost your productivity.

- **Have a clear vision**
  Make sure your automation strategy is aligned with wider business goals. Having a defined purpose behind why you're using automation will ensure it can really deliver value – whether you're improving operational efficiency, empowering employees, or ensuring process quality. And it's vital that key stakeholders are involved in strategic planning, particularly the IT team. If they see automation as a core business objective, they'll support it and make sure you have the resources you need to deliver.

- **Plan for scale**
  Think ahead about how automation can scale across the business. Will you take a centralised approach that incorporates the entire organisation or retain automation within a particular function with a central team that controls standards? If you choose the former, you might want to create a 'centre of excellence' to lead the way

in exploring and adopting new technology or practices. Whereas that latter might require delivery pods that are responsible for developing their own automated processes.

- **Fastest benefits first**

  Pick processes that automation can help generate big benefits for at pace. These are the kinds of processes prone to human error, that require a large number of manual and repetitive tasks, or that are impacting customer experience. Before automation takes place, you might be tempted to streamline a process. Consider whether automation alone might be enough, because improvements are costly and might have little impact.

- **Take your time to define**

  Don’t rush to robotics – before you even start to automate processes, gather all the right information to help mitigate any problems. That means bringing subject experts along for the ride and holding a process walk-through for the people who matter most. Make sure you document as you go and share with the team how automation will differ from the human process you previously had in place. Once everyone’s happy, it’s time to get sign-off from key stakeholders and conduct peer reviews.

- **Pick people wisely**

  You'll want a skilled and experienced lead developer to run any automation implementations, which might mean hiring new talent or working with an external partner. Make sure your people receive training where necessary—exceptional development and process analysis skills are vital. And unless you want standards to slip as the team grows, consider appointing people to oversee quality and manage the robots.

- **Deliver with care**

  Once development is underway, don’t skip any steps. Sign-off testing helps get everything up and running effectively before unleashing automation on the business. When everything’s working as you want it, don’t let the robots have all the fun. Build in business referrals or exceptions, so that the operations team can intervene where necessary or if things don’t go to plan. Automation can only work with a strong supporting IT infrastructure with capacity to support where necessary.

Continuous integration and delivery: an agile approach

Continuous integration (CI) and delivery (CD) forces developers to merge their source code updates with others' far more frequently than they might otherwise choose to, and operations teams to make sure they’re not creating a bottleneck with manual testing, to get releases out to users.

Unlike waterfall-style development, this brings conflicts and issues to light much earlier in the pipeline. Meanwhile, continuous testing (both automation and manual) ensures a high level of quality is maintained, exposing any security or reliability problems in the process. But automation alone doesn’t tick CI and CD off your list.

Here are a few other workflow and cultural implications to consider:

- **How to shift left**

  CI/CD is all about 'shifting left' – testing early to prevent rather than detect problems. It helps to increase the quality of releases and avoid nasty problems at the end of the development cycle. By working side by side, development and operations can recognise and iron-out differences in deployment – a common cause of failures in production – to create standard procedures. You can reduce the failure rate further by making all environments in the pipeline closely reflect production using cloud and pattern capabilities.

- **Sooner rather than later**

  Integration should happen early and often. Too long with one developer and you run the risk of countless conflicts when their code merges with the main repository. And if they integrate too late in the process, it's hard to understand conflicts, leading to lost time. Regular integration also allows developers to share valuable knowledge in bite-sized chunks.

- **Stories that include tests**

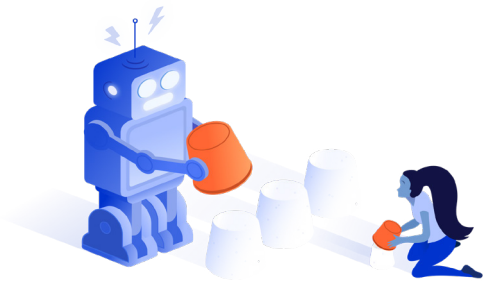
  Developers should be involved early on in user stories, helping them understand business requirements and build relationships with product managers. It also means every feature can have a plan for automated tests built in – it will cut down time spent fixing bugs on each iteration and give you confidence to make changes by quickly checking all previous features are working as they should.

- **Roles will be redefined**

  While they will still carry out some exploratory testing for complex use cases, automation means QA engineers won't need to waste time testing trivial capabilities manually. It frees them up to support developers and help build better testing strategies, tools, and datasets.

- **Overcoming auomation fears**

  Some people will be wary of CD – they'll see automation as the enemy as it's essentially taking over parts of their job. Build enthusiasm and confidence into your team with the help of experienced automation cheerleaders. If you have the right people for the job, they'll soon realise that repetitive tasks are more suited to automated tools and that automation frees them up to work on more interesting projects.



Metrics: measure your progress

Metrics provide the information you need to figure out the bottlenecks and make all your DevOps efforts worthwhile. Underpinning all these processes is the mantra 'implement, measure, improve'. There's no point having a speedy, heavily automated CI/CD process with a truly collaborative culture if you don't take the time to measure what's working and take actions to improve what's not.

There are three different kinds of metrics: time-based, quality, and automation. The first will help you save time and ship code as quickly as possible. Quality metrics, perhaps the most important of the three,

ensures that no matter the speed, quality doesn't suffer. While the third set measures the impact automation has had on your deployment process.

Together, these help to measure software performance and availability, improving stability in the process. By continuously taking measurements, you can identify causes quickly, preventing outages and minimising disruption for users. Some typical data points you might consider measuring include time to market, deployment frequency, change success/failure rate, security test pass rate, and mean time to recovery from pipeline failure.

Make more of your value stream

Metrics go hand in hand with value stream management (VSM). A value stream is everything in the delivery lifecycle – from inception to production – that's required to deliver your products to customers. A product's value is determined by its customers. To maximise ROI, each step in the value stream should create value, according to the customer's wants and needs. Processes like value stream mapping bring together stakeholders from across the business to document all the activities and hand-offs involved.

By effectively managing software and service delivery this way, you can more easily identify which steps are adding value and which aren't. As well as leveraging real-time metrics, VSM helps you break down operation silos, encourage cross-team collaboration, and coordinate and automate workflows.

Value stream management: optimise your SDLC

VSM involves the mapping, analysing, and optimising of your entire workflow (otherwise known as the value stream) from idea to delivery. It ensures that every step in the process contributes to value creation – an integral part of your DevOps efforts to deliver customer value efficiently.

It complements your DevOps processes by identifying bottlenecks, inefficiencies, and waste in your software delivery lifecycle (SDLC). It also provides visibility into processes, enabling teams to measure metrics such as cycle time, lead time, and deployment frequency as outlined above. As it's focused on the end-to-end flow, you can make improvements knowing they're aligned with wider business goals.

Like DevOps itself, VSM also promotes cross-functional collaboration. It unifies your stakeholders around shared objectives, enhancing feedback loops, and aligning technical efforts with customer outcomes. Together, DevOps and VSM can help you deliver faster, higher-quality software while maximising value and driving continuous improvement.

Site reliability practices: speed meets stability

This practice uses software tools to automate IT infrastructure tasks like system maintenance, application monitoring, and incident response, which makes managing a large system much easier. It applies engineering principles to operations, with an emphasis on proactive measures to ensure system health and performance. Essentially, it's a practical implementation of DevOps, ensuring DevOps teams have the right balance of speed and stability.

By incorporating site reliability engineers into your software team, they can set service level objectives, indicators, and error budgets based on the system's level of risk tolerance. These thresholds determine whether DevOps teams can release new features or whether they need to put changes on hold to solve existing problems. With SRE practices built into your DevOps workflows, you can enhance system reliability, reduce downtime, and speed up delivery.

Part three:

Tooling up for transformation

As teams grow and DevOps processes scale across your organisation, consistent performance is paramount. The tools you choose need to be available around the clock, and capable of performing at a high level to support your collaborative pipeline. But using DevOps tools doesn't automatically mean you're applying DevOps principles. Only by using these tools holistically with a unified culture and the processes explored in chapter two will you be able to fully transition the way you develop and deliver your products.

As you build your toolchain, you'll need to include those that step up your collaboration, keep context-switching to a minimum, aid automation, encourage visibility, and allow for effective monitoring. You might choose an all-in-one solution that doesn't integrate with third-party tools or an open toolchain that you can customise with the pick of the bunch, tailoring your toolset more closely to your organisation's needs.

Whichever approach you use, the tools you choose will correlate with the processes mentioned in the previous chapter, notably planning and building, continuous integration and delivery, metrics, operations, and feedback. Given the fast-moving nature of DevOps, your individual needs, and the rapid development of new tools, we've outlined some key considerations rather than making specific recommendations.

Build up your toolbox

- **Primed for planning**

  There's a wide variety of tools that make work more manageable and help you learn from users at pace. Whether that's work management tools like Jira and monday.com, collaborative spaces like Confluence and Slack, or DevSecOps platforms like GitLab, which helps you manage the entire software development life cycle.

  Working collaboratively is a priority, as is sprint planning and the ability to gather feedback and organise it into task-based outputs. You want effective access controls, so the right team members can share and comment on work with ease.

- **Collaborative coding**

  Get up to speed on source control and collaborative coding tools like Bitbucket and GitLab. These help you store your code in chains, giving better visibility of every change and making collaboration much smoother. Initiate pull requests to tell your team about the changes you've pushed, speeding up peer review and getting your code in the mainline much quicker.

- **Make it continuous**

  Make sure CI and CD stay top of mind with tools that make it easier to check code in a shared repository multiple times a day or as regularly as your organisation deems necessary. You'll need a production-identical environment to develop in so everyone's on the same page when they're coding and a platform that automatically applies tests to development branches and offers real-time chat alerts – providing feedback from your team.

- **Testing times**

  Automation rules the roost when it comes to testing tools – they'll help you speed up development, reduce risk, and test efficiently with minimal effort. Think UI, security scanning, and load testing all taken care of. What's more, these analyse data and produce reports that help your teams identify problem areas and course-correct as appropriate. You'll also want tools that can handle exploratory testing and orchestration, as well as wallboards that give everyone visibility over results, including operations.

- **Hassle-free deployment**

  For automated deployment, you'll need to follow good engineering practices – deploy to your lowest-level environment first before replicating up to production to point out any difference and generate standardised actions. There are a number of useful tools out there to help you automate deployment, including GitLab, Microsoft's Visual Studio, and Bamboo Server.

- **Make more of monitoring**

  For monitoring to be effective, it must be ongoing – only by recording data round the clock will you be able to understand your software's overall health. You need monitoring tools that integrate with your group chat and software development platforms so notifications, changes, and projects are all tracked in the same system. This will allow you to identify and resolve problems more quickly.

- **Figure out your feedback loops**

  You'll need multiple tools that encourage and support continuous feedback, can deliver insight from all that information, and enable everyone to have access. Harness the power of dashboards that integrate with your code repository and deployment tools and let you pull all your information together in one place, avoiding unnecessary stress prior to a release.

Big tools, big decisions

When it comes to the tech and tools available, there's a lot of choice. Figuring out what will make most sense for your organisation can be hard. In general, people pick and prefer solutions they feel most comfortable with, which can be counterintuitive to transformation. This approach can also lead to a tool binge – bringing on too much technology where integration falls by the wayside and you end up with nothing that really does the job you need. Take the time to consult, and don't lose sight of your budget – the wrong choice could put your team behind competitors, prevent agile working, and make it harder to scale.

Conclusion:
Ready for the next step?

A successful DevOps transformation rests on an in-depth understanding, wholehearted acceptance, and organisation-wide adoption of the principles and practices outlined here. DevOps is centered around the continuous improvement of software, moving away from rigid methodologies. But, this agile mindset needs to apply to the people, processes, and tools that make it happen too.

As you drive change across your teams, increasing reliability, stability, and security of your products, you'll be forced to consider whether to build up or buy in the skills you need, assessing your people differently for how they can help boost your DevOps capabilities. You'll need to take a long, hard look at your workplace culture to ensure it supports your new processes, and learn which tools will help rather than hinder your digital transformation.

You don’t do DevOps for no reason. And it’s important to keep in mind who it’s designed to benefit most – the customer. But it’s not a simple case of ‘the customer is always right’. DevOps helps organisations strive for greatness and offer the customer something they didn’t even know they needed. These key principles allow you to better understand customer pain points, working efficiently to put things right. They help you stay solutions-focused – releasing features that address real use cases and removing barriers that impact the customer experience.

At Adaptavist, we help you stay ahead of the digital transformation game with our DevOps solutions. With our strong 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 discover DevOps in a whole new light

Contact us

Adaptavist

We help organisations transform to continuous change being their business as usual. We do this by supplying technology, providing advice, and delivering change through modern, iterative approaches to development, deployment, and application lifecycle management.

Adaptavist is Atlassian's largest platinum partner, supporting more than half of the Fortune 500. We are uniquely placed to provide our experience, expertise, and insight to help your business.

Whether you want training for your team, to build a software platform for your company, or to automate your existing tooling, we can help you. If you want to unlock the full power of Atlassian and transform your business at scale, get in touch with our team today.

Visit our website to learn more

Adaptavist.com

ATLASSIAN Platinum Solution Partner US GOVERNMENT

aws PARTNER Solution Provider

monday.com certified partner

GitLab Select Partner