circleci

LEADING YOUR TEAM TO DEVOPS MATURITY

BY ROB ZUBER, CTO CIRCLECI

At this point, we're familiar with the benefits of a DevOps model of software development. DevOps can help teams move more quickly and be more nimble. Using DevOps practices makes it safer to deploy changes, and therefore helps development teams be more responsive to their markets.

But what we don't talk a lot about is that adopting DevOps is a journey, not a destination. A company doesn't just start "doing" DevOps and call it a day. The process takes time, and like any new skill, involves growing capabilities both by individuals and across an organization.

Like any new skill, the growth isn't always linear. But the good news is the kind of growth we see in DevOps organizations of all kinds is compounded: the more skills and processes teams put in place, the more ready they become to handle the issues of scale and increasing complexity that face them down the road as they grow. DevOps helps teams get more productive, and then helps them gracefully handle the growth that stems from that productivity.

This is what we mean by DevOps maturity.

Teams that adopt DevOps rely on some shared basic practices, regardless of how far along the DevOps journey they are. These include a shared responsibility for operational quality, using source control, testing, and practicing continuous integration and delivery. These lay the groundwork for a functional DevOps engineering organization. Adopt these practices, incrementally, and you'll start to see your delivery velocity pick up.

Further along the maturity curve, we start to see a few themes develop: a culture of collaboration and trust, a focus on automation and tooling, and measurement and continuous improvement. These themes form the pillars of a functional DevOps org, and will influence all the work a team does. Together, they form a virtuous cycle that edges teams toward speed, growth, and resilience. We'll go further into these themes later on in this ebook, but for now, let's explore why it's worth it to strive toward DevOps maturity.

ONE

Benefits of DevOps maturity

Speed

Ultimately, organizations further along the maturity curve can adapt to change more quickly and ship features more often. By investing in incremental process improvements, automation, and culture, they're able to ship quality software, get feedback, and respond to changes in the market quickly.

Opportunity

Mature DevOps teams are also in a better position to take advantage when advancements in tooling and technology come along. Why? Because they already have a system in place that responds well to change. If you can fail fast and know something didn't work within a couple of weeks, then you have a lot of flexibility to experiment with things that might move the needle. If not, you end up testing a tool for a year or more. Or, worse, you never try anything at all because the cost is too high. A hallmark of mature DevOps teams is their ability to roll with change, and output feedback quickly.

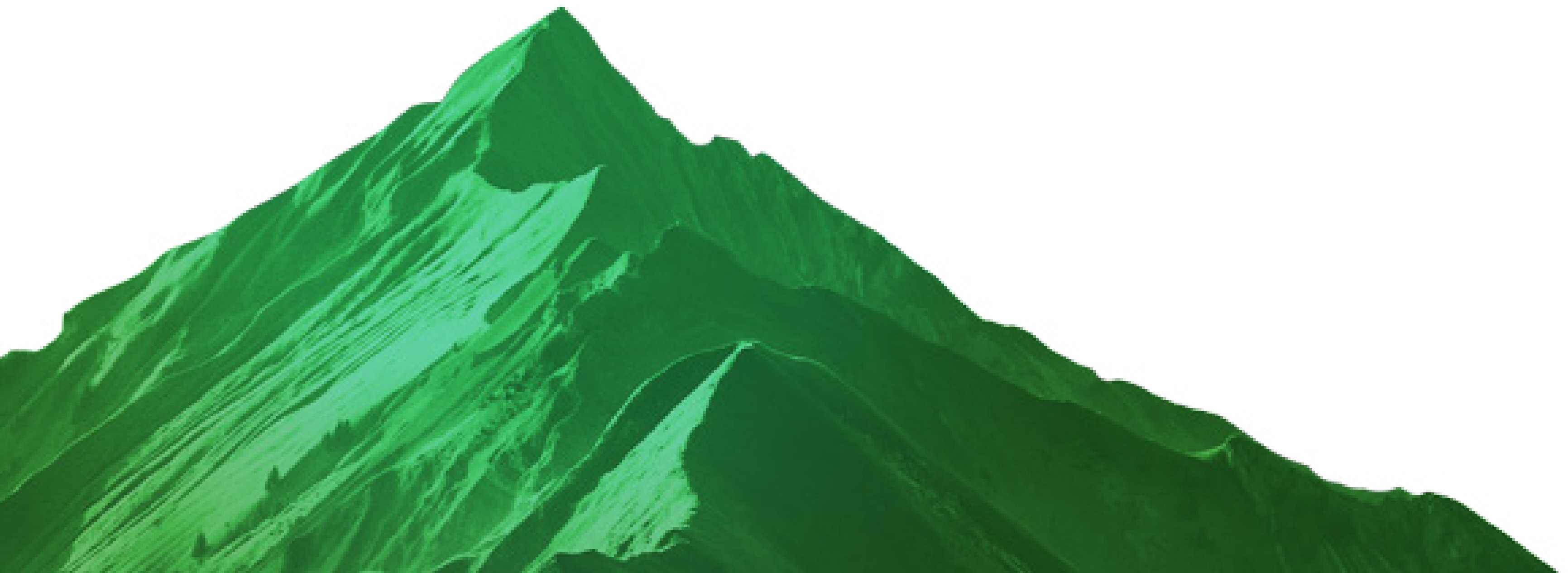
Fulfillment

Moving beyond repetitive work, to where we can focus on solving the unique problems of our business, getting value into the hands of our customers: ***this is what we all really come to work for.*** Getting to a place of cohesion and alignment just feels good. And a key part of that is eliminating the stress that used to be a part of changing the way we build. When our system is ready for change, change doesn't hurt so much.

At CircleCI, we have been working under a DevOps model for a while now, and we're still always looking for opportunities to grow. So what does maturity look like on our engineering team? It doesn't mean we know everything. It also doesn't mean we never make mistakes (because we definitely do).

Since we don't know everything, we have to test ideas often, and that means relying on the resilience of our pipeline. Using automation and continuous measurement, we can know quickly if a new tool or process is an improvement. We can implement new technologies in a controlled way: incrementally, with guardrails, and measurements.

Ultimately, being a mature DevOps organization means we're nimble enough to handle change. Our pipeline is built for it, so any shift, whether it be an unexpected stressor or a process experiment, doesn't send us off the rails. In the next chapter, I'll talk about what gets in the way of organizations moving further down the DevOps maturity spectrum, and what they can do to overcome those obstacles. I'll also talk you through building the fundamental pillars of a mature DevOps organization.



TWO

DevOps maturity blockers

Sometimes management is not yet sold on the value DevOps would bring to the organization. Among other things, they might be stuck in the “we’ve always done it this way” way of thinking. They know that no change is without risk, and even with the potential upside of increased productivity, trading known issues for unknown ones can feel scary.

Just as often, the block comes from within the team — developers are often resistant to taking on the increased operational responsibility that comes with DevOps. We’ve also heard from many operations teams who, tasked with maintaining security and uptime, fear any change that might threaten the reliability of their application.

In other organizations, management all the way up the chain of command is fully bought in on the value of the change. This team is ready to go. But both management and development teams find themselves stuck; being faced with all that needs to be done prevents them from starting. They approach DevOps with a waterfall mentality, and this desire to make a big shift all at once blocks them from making any change at all. Anything that looks too big, no one will take on — and for good reason.

If you’re the one selling up the ladder, be especially cognizant of this obstacle. There’s a temptation to want to stop all work and do a complete DevOps 180. There’s time pressure to make a change quickly in order to get early results. Combine that with a never-ending competitive push to get features out the door, and you’ll see why many attempts to switch to DevOps stall out.

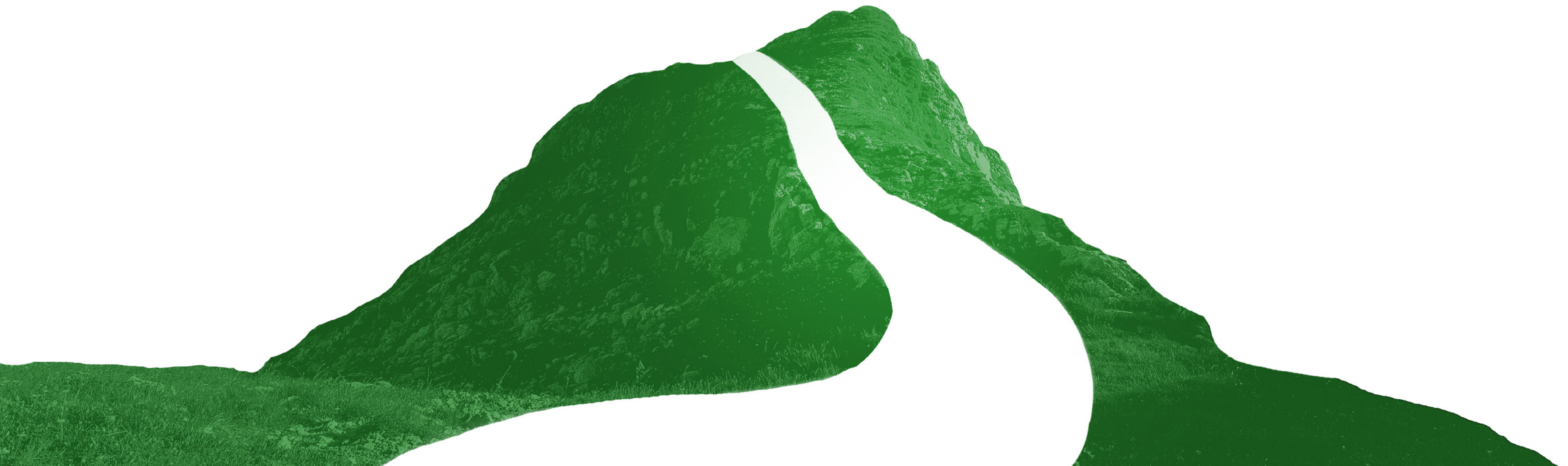
Laying the groundwork for change

Many of the biggest barriers to DevOps adoption are within and between the minds of the folks on your team — opinions and habits known together as "cultural" issues. But culture change is vague and can be a slippery objective if you don't know where to start. Making an effective change requires first knowing where resistance is coming from. While mindset problems can block change from happening, creating change at the culture level is difficult, and even when done well, is only part of the solution.

As I shared in the last chapter, there are three main pillars (including culture) that moving forward along the maturity curve requires: fostering a culture of collaboration and trust, a focus on tooling and automation, and measurement and continuous improvement.

These principles are the hallmarks of mature DevOps organizations but are useful even to teams just trying to get started. In fact, taken together, they will help you address all the blockers your team may face along the way.

For example, if management is not yet convinced that DevOps is a smart choice, you may need to make the case with numbers in order to lower the perceived risk. ***Thinking incrementally*** about the change makes it less scary, and will get you the positive results you seek sooner. Instead of asking for six months to make a full-scale transformation, ask: ***what can you do in an hour?*** Can you build a CI pipeline for one of your 50 projects, and get some early feedback? Instead of automating everything, can you ***automate one task***?



It also happened to us

It’s so easy to be stymied by a huge change looming in front of you and not know where to start. It happens to us at CircleCI as well.

Last year, our team wanted to simplify the way we expose our configuration, making it more shareable. To be honest, the project felt pretty daunting. Then we thought, “How can we pick one important use case and test that?”

We created a way for CircleCI users to write a piece of syntax and share it locally in multiple places. Then we released it to a group of users to get some early feedback. We were able to measure what people were doing in order to get more get feedback about the syntax itself. That early test led to what is now orbs, and we wouldn’t have had the confidence to invest fully in that feature without the early positive feedback and adoption.

In the next chapter, I’ll talk more about how to improve your implementation of automation, collaboration and continuous improvement, and get them working in your business.

Move up the maturity curve. Learn more about orbs, sharable packages of CircleCI configuration. Automate processes, reduce errors, and move faster.

THREE

Leading your team to DevOps maturity: The 3 Pillars

We've covered why DevOps maturity is a goal worth pursuing, how it can benefit your organization, and some common roadblocks.

Now, we're finally ready to dig into the three pillars of DevOps maturity and why they're so important. I'll also share how to get them flourishing in your organization. In case you might be inclined to choose a favorite, I want to point out that these areas are interconnected and they grow together. Each supports the other in a virtuous cycle, and when you improve one, they all benefit.

To review, the three pillars of a mature DevOps organization are:

- A culture of collaboration and trust.
- A focus on automation and tooling.
- A commitment to measurement and continuous improvement.

Now let's dig into why these are so vital.

A culture of collaboration and trust

If you don't have a culture of collaboration and trust, you might have a culture of blame and silos. Besides the fact that blaming just feels pretty bad, you've also got information hiding. When team members don't feel like it's safe to share the information they have because they fear getting blamed, then no one can move up the improvement ladder. That's a huge limiting factor. Silos are a different problem, but with a similar result: they also limit innovation by limiting access to information. Think about it this way: dev and ops used to be silos. Then we brought them together.

In a simple DevOps culture, developers are tasked with operations. This seems fine on the surface. But the problem is that all the people in the organization with experience in operations are literally sitting somewhere else. This is a silo, and learning everything from scratch is not the most efficient way of learning. We've now got a huge missed opportunity.

In an ideal scenario, when dev and ops come together, they're coming together not to do each other's jobs, but to learn as much from each other as possible. Fostering a culture of collaboration and trust means we can all solve problems faster. It means we're working together and putting all the knowledge onto the table: giving more people the chance to notice patterns, leverage historical context, and solve the problem at hand.

GET STARTED

HIRE PEOPLE YOU TRUST, and give them clear accountability. In order to create a blameless culture, you first have to have a culture that trusts that all the actors are acting in good faith, and in the best interest of the org and those around them, and they're making the best use of all the information and context they have at the time.

SHARE THE TRUTH, FIX WHAT'S BROKEN. Psychological safety is imperative for the cycle of improvement. When people fear blame, they are incentivized to hide key information, and teams spin their wheels and waste resources fixing the wrong thing.

Automation

The next pillar is automation. Any automation requires investment, and that investment requires a culture that respects time, people, and craft. Automation, therefore, means you're supporting the learning of your team in service of solving new problems. The team isn't free to solve interesting new problems until you've automated away the old problems, so investing in automation is not just a way forward to efficiency, but a way to show respect to your team members' curiosity and desire to grow. Help them get out of the weeds so they can solve tomorrow's problems instead of resolving today's.

At CircleCI, we recently released CircleCI orbs in order to make the task of authoring configuration easier, faster, and repeatable across projects and teams. Whenever possible, use readymade tools like these that automate your work (or create your own tools). Automation doesn't just save your time, but allows everyone to collaborate and benefit from each other's work. So don't stop at automating your CI, look for ways to automate your entire development process so you and your team can move on to more fulfilling tasks.

GET STARTED

MAKE THE TASK INTERESTING and enlightening. Yes, automation is partly to save time, but for people in our space, the automation itself is a craft: respect and enjoy the process, and not only will you learn, but it will make work more enjoyable. Recently at CircleCI, I was running our uptime calculations, and I decided to learn a new Python statistical analysis framework for data visualization. I did it partly because I wanted nice charts, but partly because I wanted to learn something new. Now it's a tool that I can use to do anything I want.

PLAN FOR REPETITION. The mistake we often make is thinking, "well, I'm just doing this the one time." Accept that almost anything you do is going to be repeated, and look for ways to automate those steps so you can focus on what matters.

Measurement and continuous improvement

The final pillar is measurement and continuous improvement. Continuous improvement is all about feedback. And you won't get feedback without automation (see what I meant about interconnectedness?).

Let's say you wanted to know, "How often does this process work successfully?" If it's a manual process, it's likely to be different every time, with a high occurrence of error. Automating that process lets you see how long it takes, its success probability, and exactly where problems are likely to occur. Automated systems give you reliable feedback, and that feedback is how you improve.

These three pillars working together become a virtuous cycle that drives your organization further along the maturity curve. And by practicing these tenets with your entire team, you are creating practices that scale, and learning at a much faster clip.

GET STARTED

START SMALL. Measure something you weren't measuring before. Pick a thing and get better at it. By starting small you can learn (how it works, where the pitfalls are, etc.), but you can also demonstrate. By seeing the impact, you can make better decisions about whether the impact is worth the investment and exactly where the value is.

MAKE IT DISCRETE. Much like agile processes and continuous delivery, when you have small increments of value, it's much less painful to stop. If something important comes up, you have a discrete unit of value, not a half-finished project.

FIX SOMETHING SMALL, but don't fix something inconsequential. Choose something important. Allow people to see some results from making small improvements, so you can get them to see that making small improvements is good. This will help you get people on board for making bigger improvements.

ROB ZUBER

As CTO of CircleCI, Rob helps make big technical decisions and keep teams happy and out of trouble. When he’s not testing and continuously improving with the CircleCI team, Rob enjoys snowboarding, Funkadelic and viscous cappuccino.