esper

The Beginner's Guide to DevOps for Devices

If you've been poking around the Esper site, you've probably seen the terms "DevOps" and "DevOps for Devices" thrown around. If you're not familiar with software development, modern trends in app delivery, and all that jazz, these words and phrases may not immediately click. That's why you're here, right? To learn more about what Esper does and how we're different. Good news: You're in the right place.

Without getting too ahead of ourselves here, let's get something out of the way: modern device management is incredibly complex, and every scenario is different. The needs of BYOD (bring your own device) organizations and the needs of businesses that rely on digital devices for critical functionality are very different. And that's why Esper exists — to fill the space where MDM software usually falls short and help you maximize your devices' full potential. To deliver software deployment automation (we're talking about apps, updates, security patches — the works) and the advanced device management tools that business-critical devices need. With Esper, you'll transform your devices from an operational chore to a source of innovation and engagement.

That's where our DevOps journey begins. But let's take a couple of steps back and start at the top — with MDM, or mobile device management.

Table of Contents

Mobile Device Management: A Quick Explanation

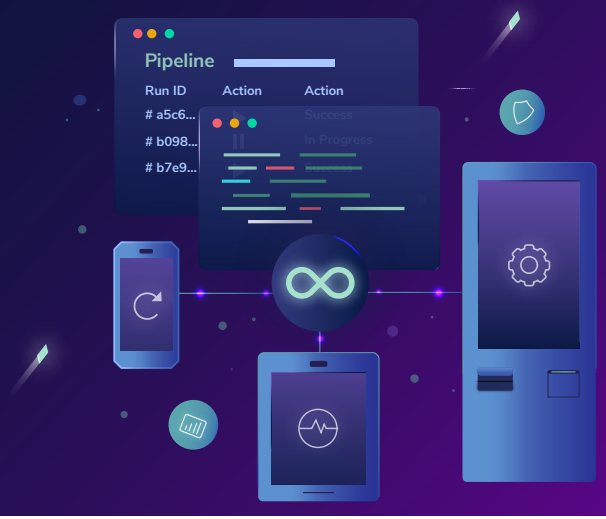
If you're here, you probably already know a little bit about mobile device management. Traditionally, organizations use this software tool to monitor and secure digital devices, but not all devices are created equal, which makes device management a challenge. That's why MDM is not the same as managing devices — one is a software tool, and the other is an action and requires the right tools (including MDM).

This is a complex topic that we could easily spend the entire page talking about, which is exactly why we have an entire, in-depth **guide focused exclusively on MDM**. It covers everything from MDM's origins to different types of device management, how the needs of business devices differ from the needs of other types of devices, and everything in between. It's a good idea to give that a read (or at least a good skim) before you start to tackle the complexities of advanced device management and DevOps for Devices.

Once you have the nuts and bolts of MDM down, it's time to get into the nitty gritty: devices. Specifically, the types of devices that DevOps for Devices is designed for.

Learn More About MDM

Company-Managed Devices Require Specific Management Strategies

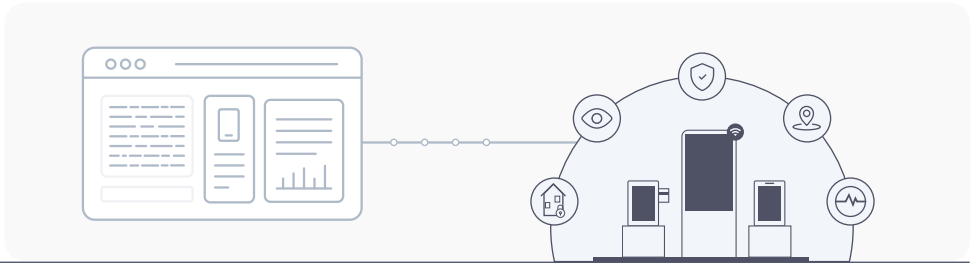


When it comes to business-owned devices, you can throw them into two buckets: the kind that are for employees and the kind that are for the business. The first bucket makes sense for COPE (Corporate Owned, Personally Enabled) scenarios, which is not the focus here. We're talking about the other bucket — the business-critical, revenue-generating, customer-facing devices that are owned, managed, and used by the business and the business alone.

Point of sale systems, digital signage, self-ordering kiosks, handheld inventory scanners, restaurant tabletop ordering systems, self-check-in kiosks, cashierless checkout systems, and airport ticket printing kiosks are all examples of company-managed devices — but this list is by no means complete. It's worth noting that these types of devices have a few names — dedicated devices; company-owned, business-only (COBO); and company-owned, single-use (COSU) are all commonly used for this category. There are some nuances worth clearing up here, so let's just get that out of the way:

- **Company-owned, business-only (COBO):** The devices are owned by the company, used by the company, and managed by the company. They can serve multiple purposes or have different roles. Think warehouse tablets or office computers.
- **Company-owned, single-use (COSU):** These devices are also owned by the company, used by the company, and managed by the company, but have a single, specific use. Point of sale systems and digital kiosks are a good example here.
- **Dedicated devices:** For all intents and purposes, these devices are the same as COSU, and the terms are used interchangeably.
- **Company-managed:** This is a group of devices that encompasses both COBO and COSU — the common thread is that they're owned by the company and used by the company. They are never personally enabled or used for anything other than business purposes.

All that is to say one thing: we're calling things these "company-managed devices" from here on out because it fits all the categories we're referencing. Simple and effective.



Devices Are Only Half the Story

Since company-managed devices are business-critical, you need a way to remotely troubleshoot, update, and otherwise interact with these devices. That's why most modern systems need a cloud backend and interface. For example, PoS (Point of Sale) systems could have a back end to configure the management of store-specific activity like user credentials, real-time transaction monitoring, and more. Logistics devices for tracking products might have a backend that manages warehouse inventory.

This information is not generally stored directly on the device, however. It's communicated back to an online repository that can be read and written from various sources across an entire fleet of devices. For example, a cloud-connected PoS could synchronize all transactions across devices in a store so management can view them in a single location.

This same kind of connectivity applies to company-managed devices. You need a way to provision devices, push updates, enforce policies (and policy changes), change configurations, and manage drift — all remotely. This is precisely why Esper exists. Whether you need an agile, upgradable point of sale system, a network of self-ordering restaurant kiosks, portable logistics devices to manage inventory, or nearly anything in between, we can help. Whatever your use case may be, Esper can help guide and enhance your device strategy with our rich set of APIs, SDKs, custom operating system, and other tools.

But you don't have to take our word for it. We can help you evaluate your scenario and decide what type of management solution will be best for you. No gimmicks, no sales. Just a quick self-assessment offers suggestions on what may work best for you.

A Good Device Strategy Starts with Your Device Management Partner

Device compatibility is one of the most important considerations for choosing the optimal device management tool. If your devices don't work correctly with your MDM, then you're paying for software that costs you time instead of saving it. Of course, assessing compatibility can be an issue since it's entirely subjective. What constitutes a straightforward MDM use case for one organization may not work for another.

That's why we put together a quick 10-question checklist to help you figure out the best management tools for your devices. This checklist will help you understand what will work for your needs and how sophisticated you need your tooling to be.

MDM Compatibility Checklist

When considering MDM for your device management needs, consider the following questions and answer yes or no. Count the number of questions you answer “yes” to on this checklist, and we’ll cover how to score your results down below.

Y/N	Questions
	Are my devices employee-owned or have a single assigned user? (i.e. not corporate-owned or for business use)?
	Do I need to lock my devices to a single application or kiosk mode?
	Are my devices responsible for generating revenue or other critical business operations?
	Do I need to update the content on my devices frequently — weekly, or even daily?
	Do I need full remote control and access to my devices from anywhere in the world at any time?
	Are my devices deployed across multiple locations, making physical access to every device a challenge?
	Am I responsible for deploying a large number of devices and need automation to scale quickly?
	Do I need remote monitoring and device telemetry and want immediate notifications if a device goes down, gets knocked offline, or falls out of policy?
	Do I need to customize the behavior or layout of an application or device interface as part of the initial setup?
	Are my Android devices running AOSP (Android without access to Google Play Services)?

Scoring the Checklist

How you answered those questions says a lot about your device use case and whether a traditional MDM tool will work well for your intended goals. Here's a quick look at how to break down your answers.

- **0:** You need basic MDM functionality
- **1-3:** You need a specific type of MDM to manage your devices
- **4-6:** Only certain MDMs can fit the bill
- **7+:** You know what you need, and not just any MDM will cut it

---

You probably realized that each question should be weighted differently for various situations. For a deeper look at why each question is important and how they may apply to your situation, you can download our MDM compatibility ebook. It's a more comprehensive look at the points outlined below, along with why each one is important and how it affects your strategy.

Download the Complete MDM Compatibility Checklist

0: You need basic MDM functionality

If you answered "no" to every question, it sounds like you have a very straightforward MDM use case. This is generally the type of situation many MDMs were made for — BYOD (bring your own device) or COPE (corporate owned, personally enabled) scenarios.

1-3: You need a specific type of MDM to manage your devices

If you answered "yes" to at least a few of the questions, then you have specific needs to simplify your device management strategy. Many MDMs offer this level of functionality, but this is where deeper analysis of each is required. You might be able to get 90-95% of what you need from a particular MDM solution, but that 5% could potentially leave you struggling at critical moments — especially as you scale.

4-6: Only certain MDMs can fit the bill

Look, you know you need MDM — that's a given. But what you need from that

MDM is pretty specific. Maybe it's powerful remote control or a tried-and-tested kiosk mode. Or perhaps it's a repeatable, scalable solution for software delivery. Whatever it is, not just any MDM provider can handle what you need. It's time to look at the fine print of what each offers.

The good news is that you're getting into the space where Esper absolutely excels. Where other MDMs will leave you wishing for something more, Esper delivers. It's why we exist.

7+: You know what you need, and not just any MDM will cut it

If you answered "yes" to 70% or more of the questions, then we were made for each other. You probably already know that not just any MDM will offer the solution you need, but maybe you're not sure where to go from there. Well, you're in the right place because Esper is the perfect next step for you.

We were designed for situations like yours. Most MDMs simply adopted the type of device management you need. But we were born in it. Molded by it.

Picking the Right Device Management Partner is Crucial

With the wrong device management tools, you're paying for software that costs you precious time instead of saving it. The same can be said if it kind of works, but it's holding you back — when you can't scale because of your management software, there's a problem.

Esper was specifically designed to address these scenarios. We saw the holes in MDM software, filled the gaps, and looked around the next corner. We're the evolution of device management, and we do this by incorporating practices found in a software development methodology called DevOps.

Instead of just focusing on the devices, our DevOps approach considers software, updates, deployment, and more — all with speed and at scale.

This is what DevOps for device management is all about — and it starts with a foundational way of thinking about software delivery.

Why is Software Deployment So Hard?

One of the biggest issues with device management is dealing with software deployment. Devices are often left insecure and left open to software vulnerabilities because delivering updates to dozens, hundreds, or even thousands of devices is just too challenging.

You do everything else to protect your company's devices and data, so leaving a gaping hole in the software seems less than ideal, right? Right. That's why you need a repeatable, predictable, and scalable software deployment strategy.

Why Software Deployment Is Challenging for Company-Managed Devices

Software deployment for app updates, security patches, and operating system updates isn't something that most people ever think about. On our personal devices, they just happen seamlessly and in the background. So what makes deploying software such a challenge to corporate hardware and other company-managed devices?

The difference is how they're managed. Security is tighter, policies are set, and devices are locked down. These devices hold important company data, so allowing every software vendor out there to update whenever they want is a big no-no. Not to mention that sort of openness would allow employees, customers, and anyone else to freely install any sort of application they want. Just imagine the potential implications. But just because software deployment is hard doesn't mean it's something you can just overlook. Quite the opposite, in fact.

Why You Need Repeatable, Predictable Software Deployment

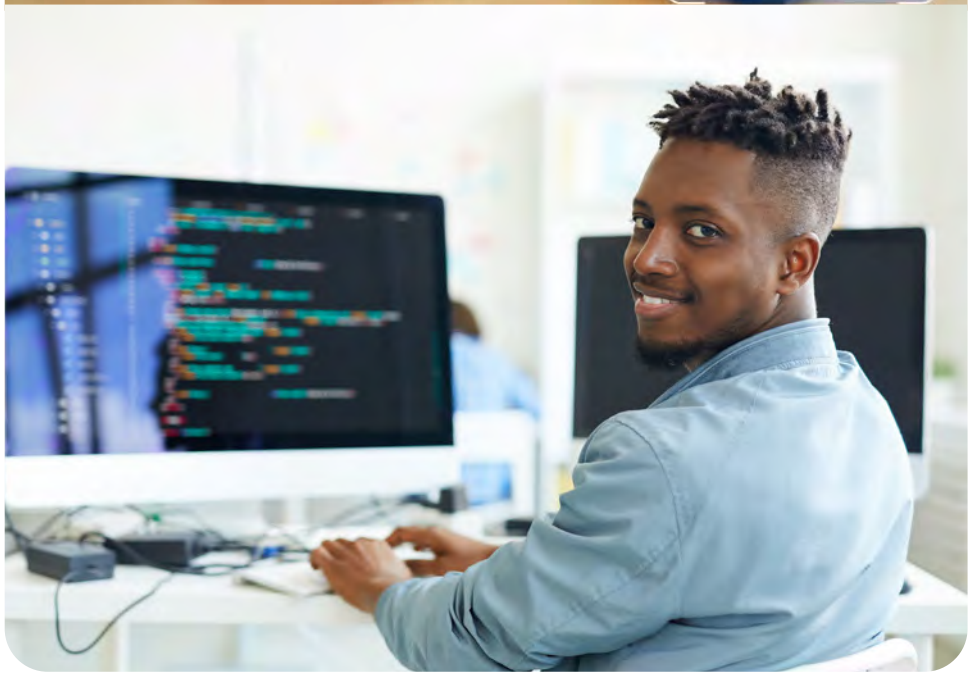
We live in an age where we rely on technology more than ever, which means "the bad guys" are working double-time to take advantage of security vulnerabilities in software at every turn. New exploits are found every day, and if you don't patch your devices regularly, they're left vulnerable. Think about the potential implications for a second.

That's why app updates, software OTAs, and security patch deployments should be simple and repeatable. Scalable. Predictable. Fun, even. (Okay, maybe not fun.) Because, let's be honest here, the easier something is, the more likely you'll actually do it. No one wants to spend three days babysitting dozens of tablets to make sure they're up to date, and they shouldn't have to.

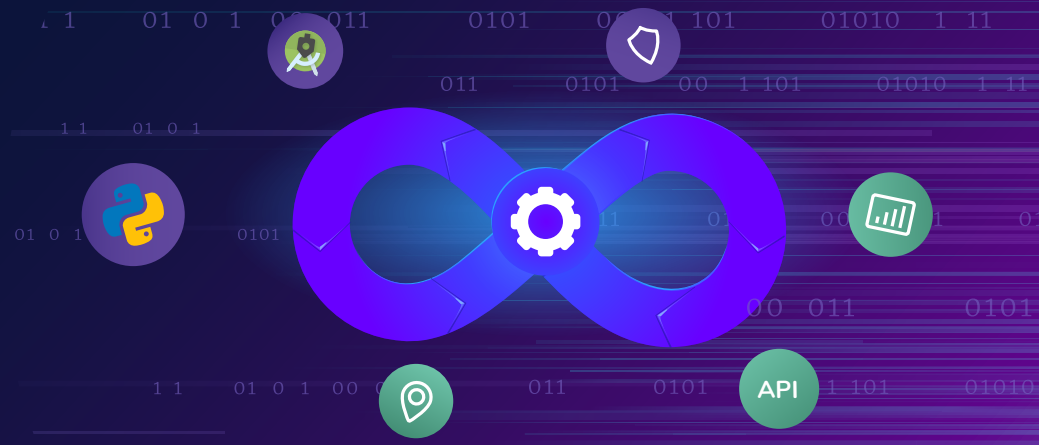
Fortunately, There's a Better Way

For years, software engineers in the cloud have been using a philosophy called "DevOps" to deliver frequent, repeatable updates to apps, operating systems, and more. That's why we took this philosophy and said, "Why stop at software development for the cloud?" and applied the concept all the way to hardware.

Let's talk about DevOps.



What is DevOps?



Semantically, DevOps is the combination of two words: development and operations. Beyond simple etymology, DevOps is a software development philosophy that aligns development and IT operations teams to deliver software in a repeatable, agile way. It's best considered a cultural shift within an organization, aligning previously siloed teams to promote seamless workflows.

DevOps practices hinge on constant communication for fast, efficient iteration. Development teams deliver new code to operations constantly so that code can be deployed immediately as a series of smaller, incremental updates instead of less frequent, larger updates. The result is a more streamlined development process that delivers continuous integration — a core DevOps principle.

But wait, what does this have to do with device management?

How DevOps Relates to Device Management

If you have company-managed devices, you also have software that needs to be managed — apps, updates, security patches, and more. Traditionally, this has been an afterthought for many companies, leading to functional and security issues. Potentially big ones.

We took everything that makes DevOps great for software development and applied those principles to hardware. DevOps uses continuous integration and continuous delivery — better known as CI/CD — where code is written and integrated into the codebase and then automatically delivered to the test or production environment. Esper takes this a step further with continuous deployment. We describe this as "CI/CD/CD," where the second "CD" describes automatically deploying tested software directly to devices in the field.

For example, let's say you need to add 150 new devices. Traditional methods would have each one manually provisioned, which is time consuming and costly.

Scale this up to thousands of devices, and you suddenly have a team of people spending days or weeks just kitting devices. With Esper, you can provision them in bulk and have everything ready to go in a matter of minutes. And that's just one example of how Esper leverages a DevOps approach for devices.

Going Beyond MDM with DevOps

When applied to devices, DevOps principles allow organizations to go beyond simple MDM practices for total control of their devices. Software deployment, OS updates and security patch installation, configuration management, and more are all simplified, streamlined, and — best of all — automated.

If you need a modern solution for not just managing a device fleet but innovating and deploying software and products at scale, DevOps is the solution. This is a concept that we built our entire company on, and it's one that we fully believe will revolutionize the way organizations, big and small, handle their company-managed devices.

We call it DevOps for Devices.

DevOps in the Cloud vs. DevOps for Devices: What's the Difference?

Getting Started with DevOps for Devices

DevOps for Devices is a relatively new concept and can confuse anyone new to device management, but you're in the right place. This is a top-level, jargon-free look at what you need to know about moving away from traditional MDM and getting started implementing DevOps for Devices.

Software Deployment to Company-Managed Devices Is a Challenge

But the right infrastructure already exists

Software deployment to devices in the field has long been challenging for organizations. Once upon a time, it involved writing software code to a CD and shipping it to the site. CDs later became USB sticks, which evolved into over-the-air updates via the cloud. While cloud deployment is far more efficient in every measurable way, it hasn't removed "the last mile" problems presented with updates to company-managed devices.

This is why DevOps for Devices was inevitable. Organizations need reliable, repeatable, robust ways to get software updates — app updates, operating system updates, and security patches — to their devices. Customer experiences are more important than ever, and the status quo isn't enough to keep customers engaged and happy. DevOps is the answer to constant iteration and improvement.

But what does that actually look like? Think about the app store on your smartphone. When a developer releases an update to an app, it's usually pushed out in stages. It starts small, and then scales accordingly as long as there are no issues. This sort of staged rollout is a foundational aspect of taking a DevOps approach to software deployment. But you probably don't have an app store to roll out your app updates. That's where Esper comes in — we enable this kind of app deployment to all your devices. You can control which devices get which app updates (and even specific versions) with absolute precision. And once you set up the deployment targets, you can fully automate the entire rollout.

Say goodbye to "the last mile" problems.

How DevOps Enables Smarter Software Delivery to Your Devices

Software deployment to company-managed devices is a challenge

You may be asking yourself how Esper works like a dedicated app delivery service for your devices. Previously, we touched on a core DevOps principle

called Continuous Integration and Continuous Delivery (CI/CD) and how Esper enables Constant Deployment (CI/CD/CD). We do this through a feature called Pipelines.

With Pipelines, you first choose which devices you want to update. This can be a small test group in your device lab, a larger group in the field, or any other combination. After you select your deployment group, you define the second (larger) group, and then a third group to hit all the devices you want to update. As long as there are no issues, your entire device fleet will get the latest software version — all automated without human interference. If there is an issue with the deployment, the whole process stops, and an alert is generated. Once it's fixed, the rollout can continue as normal. And this isn't just for app updates, either — you can use Pipelines to push operating system updates and security patches, too.

What this means for you is a device fleet that's always running the software you want.

Software Deployment Doesn't Stop at Delivery

What happens next is just as crucial

The job isn't finished once the update is in the field, of course — then it's time to ensure everything is running smoothly. You have to monitor your systems for performance hiccups like CPU spikes, increased RAM usage, battery life issues, longer response times, and all that good stuff. This combined monitoring is called observability, another (more modern) DevOps concept.

Observability is crucial in performance monitoring because it creates an endless feedback loop. It allows developers to monitor device performance, pinpoint potential causes for performance degradation, and then plan to address those in the coming release cycles. On a longer timeline, implementing observability practices will make it easier to spot performance issues before they truly become problems, thus decreasing downtime and increasing reliability. Both of those things ultimately lead to increased productivity and revenue.

Esper enables observability by offering advanced telemetry for all devices. On a single dashboard, you can easily keep track of metrics like available RAM, device temperatures, event feeds, and more. As your device fleet grows and it's impractical to monitor every device individually, we offer all the tools you need to manage by exception — in other words, only when a device encounters an issue. Ideally, observability is utilized alongside your software deplyment to catch potential problems before they affect your entire device flet.

Security Becomes an Integral Part of Your Device Strategy

And your custom experiences won't suffer, either

Security for your company-managed device fleet should be a priority, but implementing good security practices can be daunting. One thing is for sure, however — hackers are working overtime to get to your data, and breaches are under more scrutiny than ever. You absolutely can't afford to ignore device security.

That's why security compliance and testing must be integral to your device development strategy. You're already testing apps and app updates for bugs, so testing them against security updates should be part of the process. The same goes for system updates and, naturally, security patches.

With Esper's device groups and granular software rollouts, security and performance testing become one and the same. Outside of software testing and deployment, compliance policy enforcement and drift management enable IT teams to make sure devices are always operating under the most recent set of rules, are covered by the latest security patches, and continue to offer the ideal software experience for your customers and users. Win-win-win.

The Key to Scaling with Efficiency

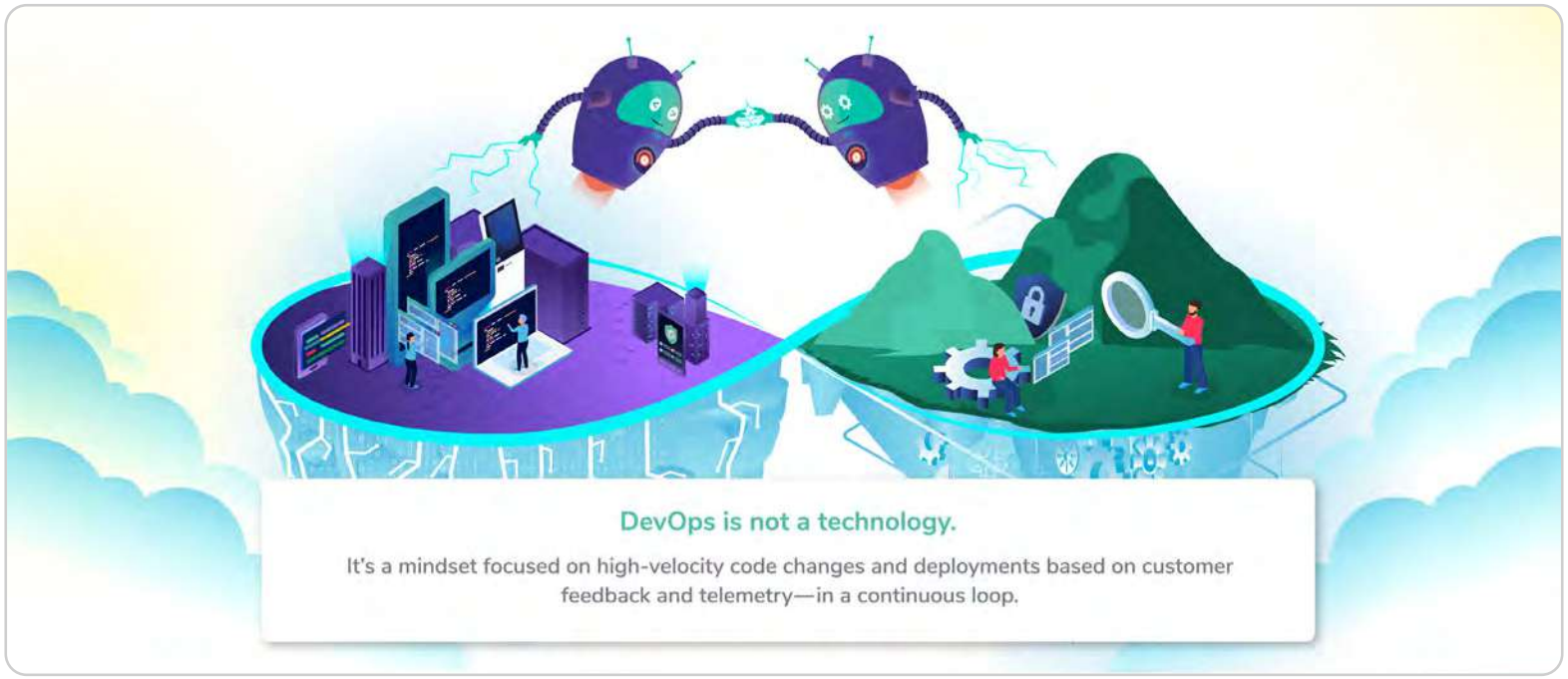
It's all about organization and process implementation

Starting with 100, 500, or 1,000 devices may seem like a lot (and it is!), but what

happens when you add that many devices at once? That's what scaling feels like to a lot of companies, and it's something you have to prepare for ahead of time. Integrating the right processes and methodologies early on ensures that you have all the right pieces in place no matter how quickly you grow.

- **Automation:**
  At scale, manual processes don't work. Integrate automation early, and you'll be more than thankful later.
- **Manage by exception:**
  When you have hundreds, thousands, or tens of thousands of devices, you can't monitor each one individually. Custom alerts allow you to manage devices by the ones that are having problems.
- **A highly organized device fleet:**
  The ability to keep track of your devices by location, type, both, or anything else you can think of is crucial to scaling. Organizing your device fleet as you build it will ensure a future where you can scale without organizational issues.
- **Plan ahead:**
  Being proactive about scaling ensures proper processes are in place. For example, custom automation on 25 devices quickly scales up to 250, 2500, and beyond. Don't wait until your device fleet is too big to manage manually to start thinking about automation.

The key to scaling is to do everything before you need to. Automate early, organize from the start, and manage your devices as if you have a much higher device count.



Tips for Integrating DevOps for Devices

It's as easy as 1-2-3...almost

That's a lot to take in, and no one goes from manual processes to full automation overnight. It takes most companies years to reach a fully intelligent device fleet that almost manages itself, but it's the steps you take today that will put you on the right path.

There isn't a one-size-fits-all guidebook to starting with DevOps for Devices because every company is on a different journey. These "rules" (they're more like guidelines, really) are designed to help you decide where and how to start — not a hard list that's meant to be followed to the T.

Prepare for friction

When you make changes, friction is going to happen. Friction from people because they've "always done it this way." Friction with existing tools because they don't fit the new approach. Friction from teams as they try to learn how to better collaborate. But that's the first step: accept the friction. Figure out where the problems are and adjust approaches as needed.

Implement better communication across teams

This goes hand in hand with the last point, and it's worth doubling down on: seamless device management is all about breaking down silos. Teams that historically may not have worked closely together will need to find ways to better communicate and integrate new processes. This takes time! Start small and work towards common goals. Just like with software deployment — trying to do everything at once is never good.

Be proactive with your approach

One of the biggest issues we see with device implementation is waiting too long to start implementing future-proof processes. When you have 25 devices, management is simple, and it's easy to do most processes manually. But even doubling that device count takes far more time for the same processes. Implement automation early, consolidate tools quickly, and streamline processes. Do it early, and revisit often.

You don't have to start from scratch to implement a DevOps approach to device management. It doesn't matter how many devices you currently have or how simple (or advanced) your current system is. Instead, look at your existing device management strategies and note the pain points. Let those define your starting point and inform your goals for tomorrow and beyond.

Continuous Improvement isn't just a core DevOps principle — it's the key to building better processes, integrations, and future.

esper

About Esper

Esper is on a mission to power exceptional device experiences by revolutionizing the way companies manage their device fleets. Through advanced capabilities, such as remote control & debugging, Pipelines for software deployment, Esper device SDK and APIs, Blueprints for dynamic configuration, and Seamless Provisioning, Esper is leading the market beyond standard MDM practices into the modern era of DevOps for devices and beyond. Recognized as one of Deloitte Technology Fast 500, Esper's innovative solutions support some of the world's most innovative brands in retail, hospitality, logistics, healthcare, education, and more.

Learn More