esper

The Fleet Planner's Playbook

This nine step guide is designed to guide you through the fleet-building process — from defining your business model to choosing the operating system and streamlining your devices with full automation, we're covering fleet management from start to finish. (Just kidding, there is no finish.)

Step 1: Defining a business model for your devices

Deciding on a business model for your dedicated devices is a crucial first step, as it will define how you move forward with your fleet — and any potential limitations associated with it. The first thing you need to ask yourself is "Do I want recurring revenue?" There are three primary models to choose from: sell and done, hybrid, and ARR-driven.

The sell and done model: *Quick turnaround with no recurring revenue.* The sell and done model offers the shortest turnaround time, but at the cost of recurring revenue. There is no maintenance or update schedule here, similar to what you would expect from a non-tech product. A good example of a successful sell and done model is a traditional analog watch — the product you start with is the same product you'll have at the end of its life.

Sell and done was pretty common in the tech products of yesteryear, but isn't a good model for a modern tech company. Customers have come to expect new features, security updates, and more from their devices, and failure to deliver on this expectation could be potentially harmful for growth. Even if your devices won't be sold to customers, this fact is still worth considering — you need a way to update and innovate on your customer or employee-facing devices.

The hybrid model: *Potential for recurring revenue, maintenance required* The hybrid model combines sell and done with the ARR-driven model (which we'll talk about below) by offering a product that is capable of receiving updates and new features, but doesn't have to in order to function. A good example of this model would be a wireless router that offers advanced features like network monitoring and security enhancements for a monthly fee. These are "nice to have" features, but aren't required in order for the router to serve its basic functionality.

The upsell features generate recurring revenue, but it's worth keeping in mind that all customers — both paid and not — will expect regular maintenance and security patches on this type of device. Planning for a hybrid product is extensive because the revenue generated by paid customers will have to cover the cost of updates for all customers.

The ARR-driven model: *Built on recurring revenue, consistent and constant maintenance required* This model essentially guarantees recurring revenue, as it's part of the overall design. Many modern connected tech companies opt to use this model as it presents the possibility of the biggest profit, albeit at the cost of constant iteration. It generally takes much longer to bring an ARR-driven product to market as existing products require constant maintenance, thus reducing the time available to create new products.

With the ARR-driven model, hardware is often sold at break even prices or even a loss in exchange for a monthly subscription. An excellent example of this is connected fitness equipment that requires a monthly subscription in order to perform most functions. This strategy requires the biggest time and financial investment to build and maintain, but offers the potential to generate remarkable profits.

Once you've decided on a business model, changing directions can be challenging. That's why it's important to commit to the right model out of the gate.

Step 2: Selecting an operating system

Once you've decided on a business model, it's time to start thinking about the operating system your devices will run. This is a big decision, as each major OS player has its own set of pros and cons. Do you want a consistent, streamlined experience that comes with a high price tag? A fully customizable operating system that requires in-depth technical knowledge to build and manage? Something in between?

Just because an operating system is "free" doesn't mean it's free

On the opposite side of iOS, you'll find Linux. This is a freely available operating system with many distros to choose from. With the right engineering team, you can even download the Linux source code and build your own custom distro specifically for your devices. That also means you can optimize Linux to run on a variety of hardware choices, from computers to tablets.

The biggest downside of choosing Linux is the maintenance overhead. It's quite complex and requires a significant investment from your engineering team to build and maintain.

PCs are ubiquitous, but offer a very limited experience

If you want any part of your device fleet to be portable, a Windows-based solution will be challenging as it isn't specifically designed for portability or touch-based interfaces. Pair that with an added expense for simply using the operating system, and things can get pricey quickly.

Touchscreen and portable devices are increasingly used for dedicated device fleets and the operating system should be optimized for both. If your device fleet will always be in a fixed location and designed for keyboard and mouse input, then Windows may be a good option for you, but if you want versatility and options, it's not a good choice.

Free, flexible, portable and adaptable: Finding the Goldilocks option

While there's no such thing as the "perfect" operating system for a dedicated device fleet, there is an option that can be "just right:" Android. As an option, Android is freely available to download and customize with AOSP (the Android Open Source Project) but doesn't require the maintenance of Linux since Google regularly patches and updates AOSP. Unlike iOS, It runs on a variety of hardware from hundreds of manufacturers, but still offers a familiar and streamlined user experience. It's optimized for touchscreen devices (but also works well with keyboard and mouse input) making it more versatile than Windows.

This is why we chose to use Android as our primary offering. Its combination of versatility, cost efficiency, simple management, and relatively light maintenance overhead, we think it's the optimal solution for dedicated device fleets of all shapes and sizes.

Step 3: AOSP vs. GMS Android

With the signs pointing toward Android as the best choice for a dedicated device fleet, there's another big decision: do you go with a custom AOSP build or an off the shelf GMS offering? There are pros and cons to each, especially when your specific use-cases come into play.

AOSP is free, customizable, and wildly versatile

AOSP — the Android Open Source Project — is the purest form of Android. It's freely available to download, modify, use, and even redistribute, which makes it perfect for many dedicated devices. That said, it's likely lacking many of the features commonly associated with "Android." For example, AOSP Android doesn't have an app store of any kind — the Google Play Store is part of Google Mobile Services (GMS, which we'll talk about in more detail below). AOSP also doesn't include things like Gmail, Chrome, YouTube, and many other Google services.

That said, if you don't need these things, then AOSP is a great option. Not only does it provide a more streamlined experience and reduced risks (you won't find random apps installed by bored employees or nosey customers if there's no app store), but you can create your ideal experience since AOSP has to be custom-built for each device. You also have greater control over the longevity of your devices since you're not at the mercy of a manufacturer or OS or security updates.

GMS is ubiquitous but more limited

If you're familiar with the version of Android that's found on the majority of tablets and smartphones, then you're familiar with GMS Android. This is Google's "version" of Android that contains all the proprietary bells and whistles, like the Google Play Store, Gmail, and even many specific features.

These things are great on consumer devices, but on dedicated devices they're often unnecessary fluff at best and potential security risks at worst. That's not to say GMS devices don't have their place in device fleets, because there could be good reasons for needing access to the Google Play Store or Chrome web browser. But if you want to lock end users out of these things completely, what better way than to never have them in the first place?

Of course, that's just a very surface-level comparison of AOSP and GMS. If you're struggling to decide, have more questions, or just want to get into the more nuanced differences, contact us. We love chatting about this stuff.

Step 4: Off the shelf vs. custom hardware

This decision goes hand-in-hand with the previous one. If you decide to go with GMS, you'll be limited to off the shelf hardware. But if you want to go with AOSP, you'll have to decide just how custom you want your custom hardware to be.

It's worth mentioning that there is a way to achieve both custom hardware and a custom OS with GMS, though it's costly and time consuming. In order to have a device GMS certified, you have to get approval from Google. This certification process takes significant time and money, but it is an option if you want a fully custom setup with GMS access.

If the return on that investment doesn't seem worth it to you, then you'll need to decide whether you want a fully custom experience or an off the shelf one.

Off the shelf is simpler, but has major drawbacks

When buying off the shelf, it's fairly easy to find the device you want and call up the manufacturer to purchase in bulk. So if you need 125 new tablets, you can likely get them in a matter of days. The downside is that you're committing to that manufacturer's security practices and update schedules, as well as their set EOL (end of life) date for that particular device. This is especially crucial to keep in mind if you're buying older off the shelf devices — even something that has been out for one year has a dramatically decreased support lifespan.

Custom hardware is more costly upfront, but puts the control in your hands

The decision to go custom isn't one that should be taken lightly, but it's absolutely worth considering. You have a couple of options for custom hardware. The first is to find a vendor selling “naked” hardware that you can purchase without an OS, then build the software to your liking. This is the most cost-effective way to go custom, even if the hardware is whitebox and the “custom” part is all in the software. are two options. First, you could work with a hardware vendor to buy readily available hardware without an operating system. You’ll be restricted to what the vendor has available. Otherwise, you can go full custom — you define the design and the hardware and have a manufacturer build it. This option is far more costly.

If you need a truly custom solution, you’ll need to find a hardware partner that can help you design what you’re looking for and build it for you. A truly bespoke option will be quite costly, but if you need hardware to match the vision in your head, it’s the best way.

Ultimately, you have to decide how custom you want the experience to be — can you get by with “stock” hardware and a custom OS? Or do you need a fully custom device? If your devices will require access to GMS, both of those questions are answered for you unless you want to deal with Google’s GMS certification requirements..

Step 5: x86 vs ARM

Oh, you thought you were done making hardware choices? Nah. When deciding on hardware — whether you decide to go off the shelf, partially custom, or fully bespoke — you need to think about processor architecture. There are two primary choices these days: ARM and x86.

Each processor type has its own set of pros and cons, so (as usually) there's a lot to consider here.

x86 is powerful, but not very portable

x86 processors, like those produced by Intel and AMD, are commonly found in Windows PCs, but have their place in other types of high performance hardware as well. Because of the increased power that these processors offer, they're typically not a good choice for mobile devices like tablets or even smartphones (or any other device that will run on battery power). But if you need powerful and fast digital signage or kiosks, it's not a bad choice.

There's also another side of the x86 coin to consider: device flips. What does that mean? It means pivoting a Windows PC (typically an older model) to Android. If you already have old Windows hardware in your organization, flipping to Android might be a great cost-saving option for you. Get in touch with us if this is something you'd be interested in.

ARM is portable, but not as powerful

If portability is what you're after, ARM is the chip for you. ARM processors are optimized for mobile devices of all types so it's highly power efficient while still being plenty powerful for most modern POS systems, mPOS, single-use kiosks, and more. If you need to run devices with complex applications, it may not be an ideal choice, but it really depends on just how “heavy” your app is. We have a team of hardware experts ready to talk about ARM vs x86 if this is an internal battle you’re facing, so get in touch with us.

It’s also worth mentioning that the GMS vs AOSP discussion has to intersect here, as well — if you’re looking to flip older Windows x86 hardware to Android, for example, you’ll have to go with AOSP.

Step 6: Building vs buying a device infrastructure

Oh, you thought you were done making hardware choices? Nah. When deciding on hardware — whether you decide to go off the shelf, partially custom, or fully bespoke — you need to think about processor architecture. There are two primary choices these days: ARM and x86.

Each processor type has its own set of pros and cons, so (as usually) there's a lot to consider here.

Building is flexible but a constant expense

For a fully customizable management solution, it's hard to beat building your own. You get full control of everything from the look to the functionality (and everything in between). But you know what that means, right? Money. Lots and lots of money. And it's not just the upfront costs, either — device management isn't "set and forget" so your infrastructure will need regular maintenance, updates, and upgrades. You'll need a dedicated team for your dedicated infrastructure.

If you already have a robust developer team, then perhaps the lift won't be as heavy for you, but keep in mind that you'll almost certainly need to expand that team.

Buying is simpler but with compromises (maybe)

An off the shelf infrastructure solution is going to be far easier to come by and significantly more affordable. So what's the downside? You get what you get. With a custom infrastructure, you can design it how you want and add functionality as needed. With an off the shelf solution, you don't get that sort of flexibility. So instead of bending your device management solution to what you want, you have to bend your uses to fit within the constraints of your management solution.

Granted, it's not all bad — some vendors will actually work with you to add crucial functionality if you need it. It never hurts to ask, right? Ultimately, this is still preferable to building something bespoke, even if you have to compromise on a feature or two.

Step 7: The app deployment dilemma

Getting apps on your device is another consideration — distributing apps to commercial devices is not the same as it is for consumer devices. Your app is one of the most crucial parts of your device experience after all — this is the defining experience you want to provide to customers or employees (or both).

Updates are important (and complex)

When it comes to delivering apps to your dedicated devices, you have choices. Are you using GMS? You could deliver and update through Managed Google Play. Are you using an AOSP-based build? You can deliver and update apps locally or over the cloud, the latter of which will rely heavily on your management partner.

So how are you going to delivery new app experiences? If you're using GMS devices, you could just use Managed Google Play. But if you're not, there's no app store to interface with. At that point, it'll be up to your device management infrastructure to provide what you need. Over the air updates are always going to be best and simplest, assuming your management partner offers the feature in a functional way (ask us how we know).

App delivery is complicated

If you want to install an app on your personal smartphone or tablet, you grab it, tap into the app store, and install the thing you want, right? Right. But unless your organization is using GMS devices, that's now exactly how it works with a fleet. App distribution is complicated, which is why you need to consider how you'll get apps onto your devices.

As alluded to above, you can use Managed Google Play if all your devices are GMS. Otherwise, you'll need a cloud delivery service to distribute apps to your devices. Choosing the right pairing that offers granular rollout options is important — you don't want to push a single update to all your devices at once, for example (that could be catastrophic).

A good app experience, both for your customers/users and IT department, is one of the most important considerations when building or upgrading a device fleet. Get it right, and your life will be much easier.

Step 8: Choosing an operational model that makes sense

Have you thought about how your devices will create value? Operational model may not seem like something you should consider when putting together a device fleet, but it definitely is. Your devices need to align with your goals.

Design, build, and test before deployment

You're probably ready to get this show on the road, but before you launch even one device you have to design your ideal solution. This is where it all starts to come together — your hardware and device types, the infrastructure, your app(s), and everything else we've covered up to this point. Everything plays a part when it comes to how you want your devices to function.

Once you have the design down, it's time to build and test it — again, before your first device ever hits the floor for use. Design, build, test, build, test, build, test...you get the idea. Perfect it.

Consider your deployment methods

With your ideal solution built and ready to go, it's time to consider how you'll deploy and provision devices. You can hand deliver every device to the desired location and set them up manually, but that's costly and time consuming. There's also the option of hiring a 3PL (third party logistics) provider to deliver and provision your devices — this saves time, but carries a high price tag.

Or you could provision devices in the warehouse, then dropship them to the desired location and hope there are no setup issues (or there's an onsite employee competent enough to get everything working). With the right device management partner, however, you can streamline the whole process — you could work with your hardware vendor to pre-install your custom OS, then define specific provisioning parameters in your device management infrastructure for a fully automated provisioning solution. Have the manufacturer deliver the hardware to the location so it only has to be shipped once, and let it provision itself as soon as it boots up. Magic.

There are a lot of moving parts and considerations to the operational model, and this is just the tip of the iceberg. Every step of the process — designing, building, staging, deploying, and provisioning — should be considered before you start moving.
and time efficiency.

Step 9: Bringing it all together with cloud tools and automation

So that thing about automatic provisioning? That's just the starting point. With your device fleet up and running, automation is your best friend. Managing a bunch of devices (we're talking hundreds to thousands to hundreds of thousands of devices here) is tough, so letting your management solution do most of the work is key.

Automation saves time, money, and hassle

If you look at modern software development, you'll see a lot of automation in place. Our philosophies are similar but don't stop at just software — we take modern software development philosophies all the way to the edge (read: your devices). Automated provisioning? Yep. How about staged app rollouts (including updates)? Yes. Need to know if a device is kicked off the network? Easy peasy.

And so, so much more.

The cloud is your friend:

Automation doesn't happen on devices or even locally on a network — it happens in the cloud. When your devices are all connected to the cloud, your management console becomes the command center for your entire fleet of devices. Whether you need to reboot a tablet across town or debug a kiosk across the country, the cloud makes it all possible. And it goes beyond simple troubleshooting too; we're talking about app updates, security patches, updated configurations, geofencing— the list is nearly endless.

All without leaving your desk.

Okay, so…that's a lot. We get it. Each of these nine steps have multiple steps of their own, and there are so many different considerations it can be overwhelming. But we know how to get you there — from deciding on an operating system and hardware specifications to full fleet automation with remote provisioning and touchless deployments, we've seen (and built) it all. Have questions? Get in touch — we have technical experts on staff ready to help.

Get started with Esper today and start managing your fleet with more agility than ever before.