# A Practical Approach to Application Modernization

Presented by: VMware

vmware®

VMware Tanzu

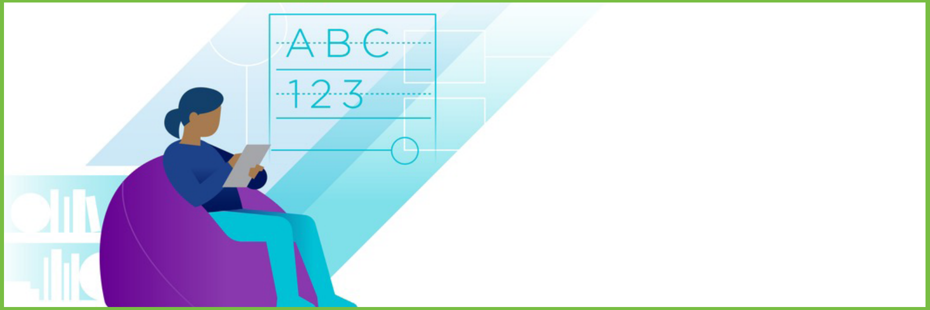
## Table of Contents

# Introduction

For many enterprises, digital transformation is taking on new urgency as a greater percentage of the world's economic activity shifts online in response to the COVID-19 pandemic. This means getting better at developing software and doing it quickly. While creating new applications is relatively straightforward, not all existing applications are built to keep pace with changing business needs, nor do they integrate easily with new apps. Modernizing your existing app portfolio has therefore become a critical part of transformation success.

For example, many companies depend on massive, monolithic applications to support e-commerce. Built on outdated frameworks and the result of organic growth over time, these applications struggle to cope with the new reality. Adding functionality in one part of the app creates regressions in other areas. Testing is complicated, maintenance is difficult, and scaling beyond a certain limit, all but impossible. Just finding people who have the expertise and interest in maintaining these apps can be difficult.

*Dick's Sporting Goods* was struggling with a number of legacy applications that were slow, fragile and expensive to maintain. These apps required heroics on the part of the IT team to operate and often failed to deliver the desired user experience—for internal users and customers. Recognizing these limitations, the company initiated a significant modernization effort in 2017, resulting in faster time to market and meaningful outcomes for its business. Throughout this eBook, we will refer to real-world examples like this one that demonstrate how companies have tackled their application modernization challenges to help illustrate what's possible.



## Who Should Read This Book?

This book is targeted to anyone spearheading an application modernization project.

Perhaps you've been assigned modernization tasks and are just looking for an entry point to get started, or maybe you are part of an organization that's struggling to deliver software features and services faster but constantly coming up short.

Either way, this book is intended to provide pragmatic, high-level guidelines to put you on a path to success. It also includes pointers to more in-depth information as you plan next steps.

Whether your target application is an e-commerce monolith, a critical financial application, or some other application specific to your business, the process of application modernization encompasses the whole range of things you can do to an application, in order to:

- Simplify management and maintenance
- Test the app more efficiently
- Ship new versions more quickly
- Scale the app more easily
- Utilize modern infrastructure
- Decrease costs

The goal of application modernization should be to deliver the specific set of benefits you need, not all possible benefits. This book describes some practical guidelines for modernization based on cloud native principles and explores important considerations surrounding culture, portfolio prioritization, and technology.

## What Is Cloud Native?

Cloud native is an approach to building and running applications that leverages the cloud computing delivery model. Cloud native is about how applications are created and deployed, not where. It's appropriate for both public and private clouds. Most important is the ability to offer nearly limitless computing power on-demand, along with modern data and application services for developers. When companies build and operate applications in a cloud native fashion, they bring new ideas to market faster and respond sooner to customer demands.

Learn More

## App Modernization: A Practical Approach

Many enterprises start the modernization process by hiring a consulting company to evaluate their portfolio of existing applications, casting a big net over everything. This approach works for convincing executives of the seriousness of the company's situation, but it doesn't provide a level of analysis that enables you to actually get to work.

VMware recommends a practical approach to application modernization that is agile and iterative. We often suggest starting with a single business unit or a sampling of applications that represent archetypes across your overall portfolio. Then, dig into these applications using code analysis to find an initial set of applications that are technically easy to modernize. Once you have at least one project going, you will start to recognize other parts of your organization and processes that need to be modernized.

This enables you to test your assumptions, understand the limits and constraints of your existing software development processes, and get a better idea of what modernization is going to require. Once you have something running, the lessons learned build confidence in your team. Being able to show tangible results in days or weeks (versus months) creates momentum for future work, and an expanded roadmap of projects.

For example, *Travelers Insurance* decided to focus on just one small—but critical—piece of its application portfolio. Focusing on a single element helped the Travelers team build skills and confidence while demonstrating they could deliver significant value.

### Principles of Practical Application Modernization

- Start from where you are today
- Start small
- Think agile and iterative
- Learn by doing
- Build confidence
- Demonstrate results quickly
- Forget about perfection

## Essentials for App Modernization Success

Regardless of your industry or the specifics of your application portfolio, there are a few guidelines that—based on experience working on thousands of engagements—we believe are essential for success. In the sections that follow, we'll explore some important guidelines in three areas:

### Culture

Culture is critical to the success of app modernization and cloud native development. Technology is great, but you won't get very far if your people aren't bought in or your organization isn't structured to deliver.

### Portfolio

You may have hundreds of applications that need attention. How you set priorities and where you start has a big impact on where you end up.

### Technology

There is no single technology stack or set of tools that ensures success. Keeping an open mind and choosing technologies pragmatically will get you further than trying to pass arbitrary purity tests.

## Culture Is Key

Much has been written about the critical importance of culture to software development success. This section focuses on three critical elements: buy-in, setting expectations, and organizational structure.

### Get Internal Buy-In

Before you start on any serious app modernization effort it's essential to make sure that you have internal buy-in for the effort—both from the top down *and* the bottom up. Application modernization necessarily distracts from other work, and that can mean decreased productivity in the near term. Both executives and developers may have strong feelings about that. When it comes to convincing people, there's no replacement for understanding the leverage points within your company.

#### Executives

The most successful modernization efforts have strong executive support, and the executives that actively support modernization reap the largest rewards. Active support means empowering people to challenge existing norms and patterns—and making sure modernization efforts don't get entangled in "that's the way it's always been done" thinking.

Executives almost always think about the bottom line, so it's important to help them understand how modernization will make the company more successful and more profitable. Most executives in established companies are now tuned into the idea of digital transformation and the importance of a rapid cadence of new software features and digital services. If you can convince a CIO that the average development time for new features can be reduced from two months to two weeks, you'll probably get their attention.

#### Developers

Application developers are motivated by shipping features and making software more viable; their applications are often on the critical path for profit generation. It's asking a lot to tell developers to stop what they are currently doing and learn a whole new approach to software. Developers have to be convinced that modernization will be worth the effort. Yet, most developers want to develop modern applications and ship faster; they struggle to do that with legacy software. The desire is there but the tools and techniques are lacking.

## Developers (continued)

The most important thing is to understand the pain points associated with the way developers currently operate. That means spending some time to understand what's going on. For example, perhaps it takes a developer four hours to test a simple code change in a valid environment because they have to open a ticket to spin up a virtual machine to do the testing. Developers by and large don't like wasting time.

By understanding their pain points, you can sell developers on the aspects of modernization that will benefit them or make them more successful. In the process, you may be able to prioritize changes that deliver the biggest returns for the least effort.

## Set Expectations Appropriately

It's important to set expectations appropriately with all stakeholders about the expected rate of progress. Making meaningful progress modernizing hundreds or thousands of existing applications will take significant time and effort.

Changes happen incrementally; modernization often starts small and grows rapidly. It's important to realize this, set expectations appropriately and take a systematic approach. It can be helpful to establish visible intermediate milestones to maintain momentum. For instance, you might want to set a goal to modernize ten things and make sure the list includes different items to benefit various stakeholders, so everyone sees some benefit right away.

If you have a massive portfolio, you simply won't be able to do everything overnight. For example, Air France–KLM relies on a large portfolio of apps to run its daily business. The company identified over 2,000 applications that could benefit from being modernized. Putting together a strategy to do this is, understandably, an undertaking that can take years. And, as you get further along, it's common for the very hard tasks to be the ones that remain.

### Consider a Tiger Team or Center of Excellence

Many organizations create a team dedicated to spearheading app modernization across the company and evangelizing and teaching new software methods. *Dick's Sporting Goods* created an in-house application transformation team to spread tools and best practices across the company, helping product teams build and deploy more quickly.

Modern software methods, built-in monitoring, and blue/green deployments make it easy for teams to identify and fix issues in minutes, without downtime.

## Restructure Teams as Necessary to Support Your Modernization Efforts

If your teams are currently developing and running monolithic apps using traditional infrastructure and manual methods, the way things are structured today may not make sense as you progress. Unfortunately, there's no single one-size-fits-all team structure that will satisfy your needs as you modernize, and just because an organizational structure worked for Netflix or Target, you can't assume that the exact same structure will work for you. In many cases, the best approach is not to start with too many preconceived notions about organization. As the needs of your organization reveal themselves, decisively adapt your team structure to support those needs. The best organizational structure for you often reveals itself over time. Well run software organizations know this and are always looking for new ways to operate, rather than sticking to one operating model.

One thing that helps as you start your app modernization efforts is to have champions across all the relevant teams, from your development teams to infrastructure teams to operations (assuming you have separate operations teams) and extending into business units as well. A few well-placed people—with the ability to recognize problems, empathy to work with people with diverging views and passion for your modernization vision—can help ensure that everyone is bought in and that things move forward.

### Creating Modernization Champions

VMware Tanzu Labs helps clients motivate team members and create modernization champions by encouraging organizations to create a team, choose a logo and foster engagement. Finding meaningful ways to rally the troops can turn many fence sitters into advocates.

### Culture: Where to Learn More

- *So, you want to build a "Silicon Valley-like" software developer culture? Focus on the people.*
- *How organizational culture can drive digital success*
- *How Liberty Mutual made a culture of constant education central to its digital transformation*

## Portfolio

If you've got hundreds of applications, it's critical to spend your modernization effort wisely. That means analyzing your portfolio and thinking critically about where and how to start.

### Making Sense of Your Portfolio

When you begin to consider your portfolio of applications, it's important to recognize that every application isn't going to receive the same treatment. Consider evaluating each application in terms of its business value and the technical feasibility of modernizing it, then focus most of your effort on the set of applications that have high business value and are feasible to modernize.

If you consider a portfolio of 100 apps, they typically fall into three buckets:

### No Change

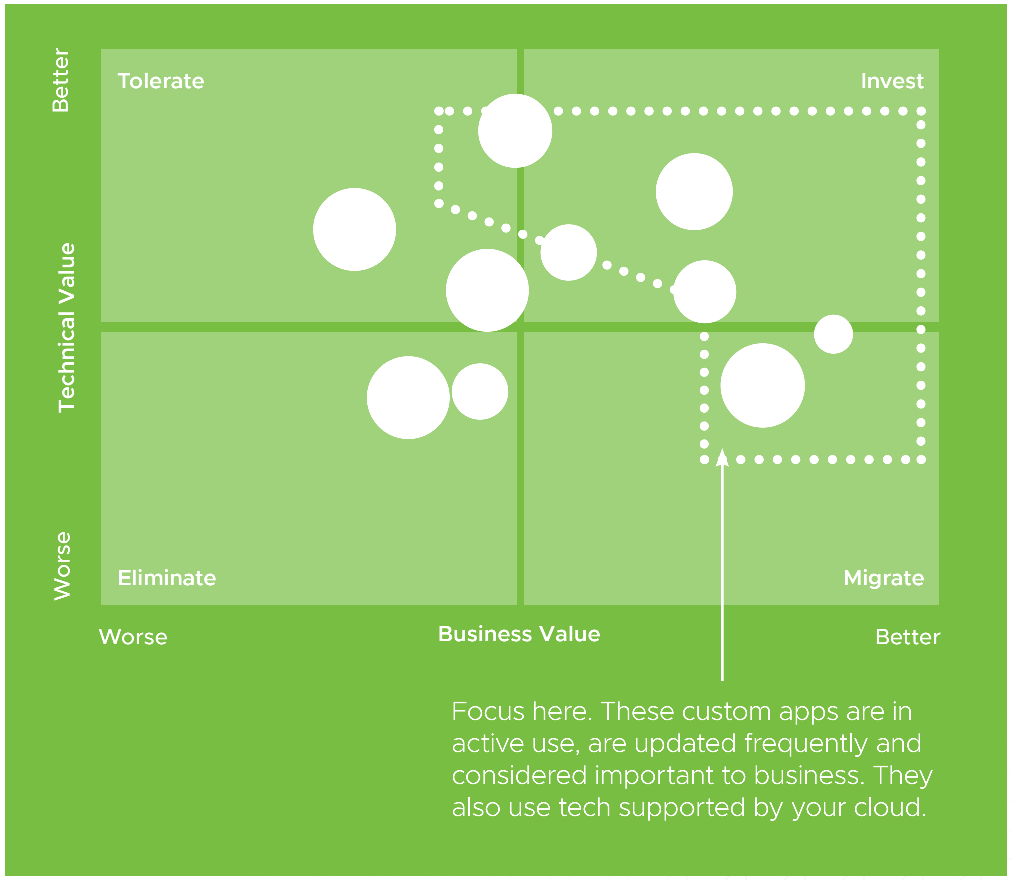
Roughly 15% stay as they are with no changes. This may include apps that are scheduled for retirement and apps that are so locked into proprietary hardware and software that they can't be migrated.

### Lift and Shift (Replatform)

Perhaps 40–50% of applications can be lifted and shifted into containers. The goal with these is often to gain automation and time-savings benefits by incorporating continuous integration and automating the path to production.

### Modernize

Just 15–20% of applications will likely receive extensive modernization. This is most often the set of applications with the greatest potential to move the needle when they are modernized.



Focus here. These custom apps are in active use, are updated frequently and considered important to business. They also use tech supported by your cloud.

## Choosing the Right Starting Point

Even with a good understanding of your application portfolio, it can still be difficult to decide where to start. There's no single strategy that works for every organization. It's helpful to start as small as possible, while working on real projects that you can learn from. For retail businesses, the ubiquitous store finder application can be a good place to start. It's useful and self-contained, so the impact is low if initial iterations don't work as expected and it doesn't require customer data (other than location) so compliance isn't an issue.

Many companies prefer to start with changes that have immediate visible impact. Travelers Insurance chose to modernize its ratings engine because it is essential for quoting competitive insurance rates. The decades old rating engine was part of a monolith running on a mainframe. This made it difficult to scale, expensive to run and reliant on a shrinking pool of engineers. However, the rating engine came with significant constraints. Performance—starting from the first production release—had to be at least as good as the existing system to avoid breaking other dependencies, and the output of the new engine had to be identical for the same reason. Having many points of integration created challenges for a team just learning the ropes, but in a short period of time, they overcame the technical challenges and successfully modernized key functionality that the business depends on.

Dick's Sporting Goods started its modernization efforts by focusing on just two product teams, creating a catalyst to disseminate knowledge and skills across more teams and more of the company's portfolio.

### Modernizing Your Portfolio: Where to Learn More

- *App Modernization 101: An Executive's Guide to Shipping Better Software*
- *App Modernization Recipes*
- *Application Modernization Should Be Business-Centric, Continuous and Multiplatform*
- *5 Tips for Modernizing Your Legacy Application: Lessons Learned at Discover*

## Technology Choices

This guide discusses technology after cultural and portfolio considerations, not because technology is not important, but because it's critical to have the nontechnical underpinnings in place first. When it comes to technology, there are proven patterns at all levels of the stack. Implementing them isn't necessarily easy, but it's mostly a matter of learning and executing. Choosing good tools enables you to do the work but doesn't do the work for you.

It's not our intention here to evangelize any particular technological orthodoxies, practices or tools. Keep in mind: when it comes to modernizing established applications, your future state is likely to be a combination of old and new technologies for the foreseeable future. Some backend systems may not get touched for a long time to come. You may simply lift and shift some application modules, while others get decomposed into microservices. Some app components may run on a public cloud while others stay on premises due to data gravity.

### Important Modernization Technologies

**Containers** encapsulate an application in a form that's portable and easy to deploy. Containers can run on any compatible system in any cloud. Containers consume resources efficiently, enabling high density and utilization.

**Kubernetes** makes it possible to deploy and run complex applications requiring multiple containers by clustering physical or virtual resources for application hosting. Kubernetes is extensible, self-healing, scales applications automatically and is inherently multi-cloud.

**Microservices** architecture breaks down an application into multiple component services, enabling greater parallelism during both development and execution.

## Making Technology Decisions

When choosing development and deployment tools, you should consider your application modernization needs holistically across multiple projects. Where possible, choose frameworks and CI/CD tools that work with the other tools you have and that are as compatible as possible with the existing capabilities and skillsets of your team. This will allow you to maximize the benefits of learning and build expertise more quickly.

Individual applications will typically require some specific architectural decisions. Choose the architectural elements that make sense for your application and your organization rather than worrying about conforming to the full list of cloud native precepts.

*Travelers Insurance* decided to develop its rating engine using .NET Core 2.x and *Steeltoe*, an open source project for cloud native .NET microservice applications; the company built out a CI/CD pipeline for deployment from day one. The chosen application architecture used a *choreography pattern* and the team also implemented a new API to make it easier for future applications to interact with the rating engine.

## Important Modernization Technologies (cont.)

**CI/CD** (Continuous Integration and Continuous Delivery) software creates an automated pipeline to integrate, test and deploy code changes.

**Messaging** technologies provide methods for passing requests between microservices, with the ability to handle large amounts of data at large scale.

**An API Gateway** can provide a single entry point into a software system for all requests, making the environment more structured and predictable. As long as the API stays the same, underlying services can be changed and enhanced.

**Observability** allows developers and operators a better look into modernized applications and helps pinpoint issues if something goes wrong. Metrics, logging and tracing help teams make sure an application is running as it should.

VMware Tanzu

## Breaking Down Monolithic Applications

Breaking down monolithic applications into separate microservices is often an important part of modernization. Having separate microservices allows you to parallelize continuing development and also provides more clearly defined areas of responsibility. Breaking down monoliths with microservices also enables you to incrementally make changes while not interrupting business as usual.

### Don’t Overdo It

Be careful not to go too far when decomposing an application. At one point, Travelers decomposed its rating engine into 13 separate microservices before ultimately settling on five services as the optimum for delivering the right combination of modularity and performance.

It’s common to get into the mindset of “we need to make this application cloud native.” But, if you look at the full set of cloud native principles, a lot of them may not benefit your application. Focus on the ones that will.

### Deciding Where to Divide

Every application is different, and only you can decide where the dividing lines should be drawn. One guideline is to consider the network as the new application interface. Anytime an application transfers large, complete data structures, that’s a possible break point.

## Swift Methodology

VMware Tanzu Labs uses the Swift methodology, a set of lightweight techniques based on agile and *Domain Driven Design* (DDD) principles, to help teams get started with incremental modernization of monolithic apps.

Swift aligns business leaders and technical practitioners, resulting in an architectural plan that maps future goals. Information gained through the use of Swift informs decisions on how to organize development teams and prioritize work (from both a business and technical perspective). It’s also helpful as a catch-all way to define a path between the status quo and the desired state.

Learn More

## Sometimes Monoliths are Okay

If you're having trouble deciding how to break up a monolith, it could also be a sign that you should leave it more or less as is. Many monoliths can be containerized and run just fine, and you can still take advantage of the CI/CD pipeline to streamline development and deployment. That may be all the modernization you need.

## Backend Databases

If your application relies on a backend database, such as SQL Server or Oracle running in a VM or on bare metal, it's perfectly acceptable to leave it running as is. If it ends up creating an operational problem down the road, these databases can be containerized, and there are a variety of open source alternatives.

## Do I Need Kubernetes?

It's become common to consider Kubernetes as essential for cloud native development and application modernization, but that's not always the case. Although VMs don't have the same set of orchestration and scaling features, microservices can run inside VMs rather than containers.

If you're an application developer, the platform choice may not be up to you or you may not care that much. If you do have a choice, pick your platform based on the needs of your application and the skills of your organization. If the application can run equally well in containers and VMs—but your organization doesn't have expertise with containers and Kubernetes—it should be a pretty clear choice.

## Should I Move My Application to Kubernetes?

The decision to move an application comes down to three factors: value, risk and time.

It's up to you to evaluate each application to understand the potential value of Kubernetes for that application.

Migrating an application to a new platform always entails technical risk and cost. Kubernetes is young as IT technologies go, as are the associated tools.

The migration effort can take significant time on the part of developers and operators.

Before you jump in, look for ways to concretely measure the gained value, understand the amount of risk and determine how much time you can afford to spend.

## Choosing Technology for App Modernization: Where to Learn More

- *Kubernetes for Executives*
- *Kubernetes for Operators*
- *Kubernetes for Developers*

## Your App Modernization Checklist

**1** Get Buy-in.
Make sure all the stakeholders for an application are brought into the modernization effort.

**2** Set Expectations.
Provide as much visibility as possible into the time and effort a modernization project will require. Avoid over-promising and under-delivering.

**3** Restructure When Needed.
Prepare for your organizational structure to evolve as modernization efforts advance. Pay attention to how other companies have organized, but don’t just assume the same approach will work for you.

**4** Prioritize Your Portfolio.
Analyze your applications and divide them into three buckets: no change, lift-and-shift (less effort), modernize (more effort).

**5** Choose The Right Starting Point.
Pick one or a few small(ish) projects that will help you start on the right foot in terms of skill-building or momentum-building, or both.

**6** Make Smart Technology Decisions.
Don’t choose a set of technologies simply because it’s what “the cool kids” are doing. Make sure your choices are right for your organization.

**7** Break Down Monoliths.
Plan carefully to decompose monolithic applications into more manageable pieces without worrying about satisfying any cloud native purity tests.

**8** Pick Platforms Pragmatically.
Base cloud and platform choices on the needs and capabilities of your organization.

### Application Transformation with VMware Tanzu Labs

Migrating existing apps can be daunting—but it doesn’t have to be. VMware can help you develop a tailored app modernization strategy that builds on many of the principles described in this eBook. Improve agility while reducing costs and tech debt with our proven Cloud Native Readiness techniques.

- *Application modernization*
- *Modernization recipes*

## What Should I Do Next?

If you're just getting started on your app modernization journey, the most important thing is to take a practical approach to improvement. Focus on the areas that you believe will yield the biggest benefits as you make cultural, portfolio and technology decisions, and don't be afraid to try things that may not succeed. Even seemingly small improvements sometimes relieve bottlenecks and yield bigger-than-expected benefits. Use the resource links in this eBook to learn more about areas where you need more guidance.

Many enterprises seek outside assistance to help kickstart app modernization efforts. Tanzu Labs helps organizations around the world modernize business critical applications and large systems across entire portfolios. We'll help you develop a sustainable roadmap for your modernization initiative. Our consulting services upskill your teams as we work collaboratively and align new delivery methods with your existing apps in a way that results in lasting improvements.

*Schedule a free, virtual consultation with the Tanzu Labs team today.* Each session is tailored to your unique business or technical challenge. We'll make sure you leave with actionable items that will make a difference.

Reserve Your Free Consultation Today

And be sure and follow *@VMwareTanzu* on Twitter to keep up with all the latest cloud native developments.

### Where to Learn More

- *Tackle App Modernization in Days and Weeks, Not Months and Years*
- *VMware Tanzu Blog*. Read our regular blog to find out the latest. Posts cover diverse topics and new blogs are added regularly.

### Customers Mentioned in This eBook

- Travelers Insurance: *Customer story* and *webinar*
- Dick's Sporting Goods: *Customer story* and *webinar*
- Air France–KLM: *Customer story* and *webinar*

### Additional Customer Modernization Stories

- Steelcase: *Customer story*