OctopusDeploy / **blue-papers** Private

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

migrating-to-c...

blue-papers / migrating-to-the-cloud / **index.md**

**terence-octo** Update index.md

1 contributor

| title | author |
|---|---|
| Modernizing your applications by moving to the Cloud and Cloud Native technologies | terence.wong@octopus.com |

## Introduction

Modernization has always been an essential factor in the survival of businesses. Depending on the time, modernization could be using floppy disks, setting up company emails, installing bare metal infrastructure, or developing a mobile application. Though the implementation may change as technology evolves, modernization requires businesses to keep up or risk being outdated.

Today's trend is modernizing applications, moving to the public cloud, and shifting to cloud-native technologies. Businesses that don't modernize experience pain points from having out-of-date technologies, including:

- Technologies falling out of support
- Difficulties finding and retaining engineers when the business is using old technology
- Increased maintenance costs
- Loss of business due to customers wanting modern technologies
- Challenges scaling solutions to meet demand

Gartner[1] predicts that cloud-native platforms will be the foundation for more than 95% of new digital initiatives by 2025 - up from less than 40% in 2021. The need to shift towards the cloud is felt universally by all businesses. Cloud computing helps you build scalable, always-on applications through distributed networks. Companies that leverage cloud computing have a business advantage over traditional methods. However, there are many barriers that companies cite to getting started, such as:

- Cost concerns
- Security
- Lack of skills
- Governance
- Culture

> The path to a successful modernization project is to do it at your own pace. Not every company can be at the cutting edge and modernize every application simultaneously. Many companies rely on legacy systems and will need a gradual hybrid approach toward complete modernization. Every journey to the cloud will look different, and there are ways to approach it no matter your stage. This white paper will help you find ways to get started on your modernization journey.

We see migration to the cloud happening in three stages, strategy, implementation, and operation. These stages agree with the consensus best practices described by the major cloud platforms, Amazon Web Services (AWS)[2], Microsoft Azure[3], and Google Cloud Platform (GCP)[4]. These stages appeal to cloud engineers and cloud managers in different ways. It is essential to understand how to appeal to both types of people to get full support for your cloud migration efforts.

We want to help you get started on your journey to cloud-native and modernization. This white paper has three parts:

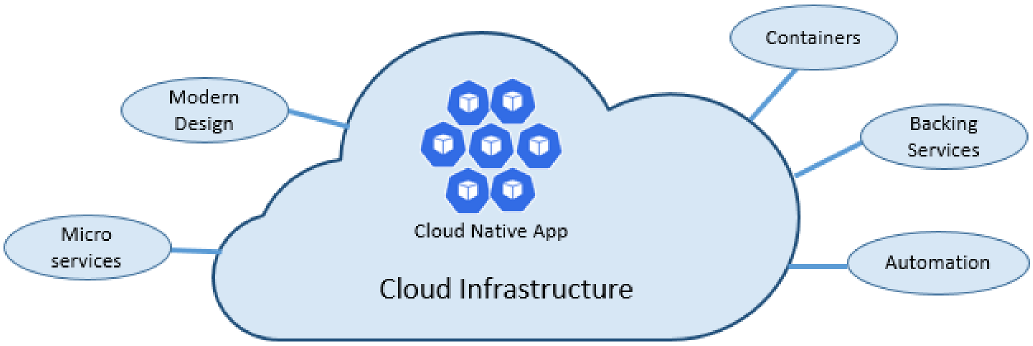
- Part 1: What is cloud-native, and why is it important?
- Part 2: The journey to cloud-native: the main cloud concepts and how to get started
- Part 3: Cloud-native and Octopus Deploy: a case study and our recommendations

This whitepaper will help you understand cloud-native and how to start your modernization journey, and the importance of modernizing at your own pace. Even small steps can build momentum for more significant wins in the future. We'll share our best practice tips and the first steps you can take towards modernization.

## Part 1: What is cloud-native, and why is it important?

There are two common ways of categorizing technology in the cloud: cloud-enabled and cloud-native. Cloud-enabled applications were designed for a conventional data center but now run in the cloud. The application could be legacy software that developers migrated to the cloud. These applications often run on cloud infrastructure rather than on modern cloud platforms. By contrast, cloud-native technologies are built for the cloud from the ground up. These applications run on a cloud by design and take full advantage of the benefits of the cloud. The journey towards modernization will involve a mixture of cloud-enabled and cloud-native technologies. Not all components in a system need to be completely rearchitected to be cloud-native immediately, however that can be a longer term end-goal.

Microsoft[5] describes the benefits of cloud-native through the five cloud-native pillars:



- **Microservices:** Microservices is an architectural pattern for developing modern applications. Microservices separate application components into smaller independent services. These services are combined to form a larger application.
- **Modern Design:** The twelve-factor application methodology[6] is a widely accepted list of principles that describe modern cloud application requirements. There is an emphasis on portability, scale, and agility to ensure that applications work across environments.

- **Containers:** Containers are portable computing environments. They are independently run and contain all the necessary components to run in isolation. Developers load applications onto containers through container images, and you can run them on any container-compliant infrastructure .

- **Backing Services:** Cloud technologies create data at high volumes and will need to be stored somewhere. Storage options include relational databases, document databases, analytics, message queues, monitoring tools, etc. Cloud platforms provide managed backing services at scale; you only need to select and provision the services you need rather than create them yourself.

- **Automation:** A primary feature of cloud-native systems is automation. You can automate many tasks in a cloud-native system, such as workload orchestration, logging error messages, or backup databases. You can use concepts like Infrastructure as Code (IaC) or Configuration as Code (CaC) to represent your system as code so you can replicate it in another environment.

A fundamental shift in designing cloud-native systems is the way developers treat resources. In traditional data centers, compute resources are cared for. They are named and nursed back to health. In cloud-native systems, compute resources are expendable and replaceable; if one is unhealthy, another takes its place. The ability to provision new, healthy infrastructure in a cloud environment instead of restoring a broken server allows DevOps teams to recover faster rather than scale to meet demand. The ability to incrementally provision new resources in the cloud instead of the large purchase of new servers in an on-premises environment gives cloud-native technologies scalability.

## The benefits of cloud-native

There is pressure on businesses to transition from legacy systems to cloud-native solutions to keep up with the competitive benefits of modernization. The State of Application Modernization Report 2022 by Konveyor suggests that reliability, scalability, and security were at the top of manager expectations for moving towards cloud-native. These benefits are thanks to cloud technologies like containerization[8] and microservices[9]. Here are some additional benefits that you can achieve with cloud-native technologies:

- **Global reach:** A cloud solution will have service uptime guarantees listed in the contract. These are known as service level agreements (SLAs). For example, Amazon EC2 provides a 10% service credit if the monthly uptime for a same-region EC2 is between 99.0% and 99.99%. Cloud platforms have data centers distributed worldwide and have a clear advantage over hosting your infrastructure in a central location. If one zone experiences downtime, cloud platforms can move the application to a nearby zone.

- **Scalability:** Cloud platforms allow you to scale resources up or down depending on demand. If resources are idle, cloud platforms will shut them down, and during peak times, they can allocate more resources to accommodate demand.

- **Security:** Cloud platforms contain inbuilt security features to handle multiple roles and access conditions. All major cloud platforms adhere to data handling requirements, and migrating to them provides you with an ISO-compliant foundation to build on.

- **Cost Savings:** Due to the economies of scale, cloud solutions can provide cost savings over traditional deployments through dynamic scaling. You can use cost calculators to estimate your total billable cloud compute upfront before you commit to implementation. A case study by Amazon showed that Brightline reduced cost by over 70% by using AWS.[10]

- **Self-Service:** Cloud solutions can offer self-service to your customers through software-as-a-service platforms. Your customers can access what they need in your application without having to interface with support. When designed well, this can lead to increased customer satisfaction.

- **Automation:** When you use cloud computing for your workloads, you can automate the system's operation and infrastructure management through cloud platforms.

- **Continuity:** You can back up your data in the cloud and repurpose the applications you develop in a cloud platform for reuse. Cloud solutions give you business continuity as you build on your previous work.
- **Latest Updates:** When you migrate to the cloud, you will receive automatic updates to all your cloud components. You get the benefits of continual improvements and innovation to the systems you frequently use.
- **Collaboration:** In a cloud environment, it is easy for your developers to collaborate on a shared

546 lines (323 sloc) | 67.7 KB ...

- **Cloud Familiarity:** Familiarity in the cloud can also be a competitive advantage to your business. If you have moved to the cloud, you can hire people who already have the required skills.

## The journey to cloud-native

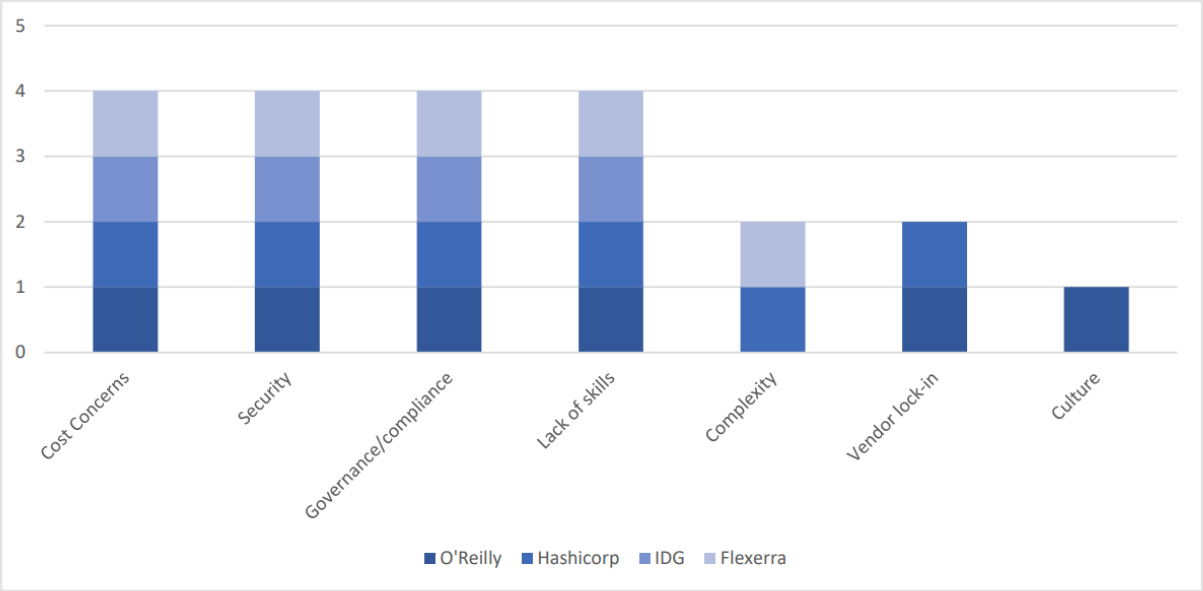
Perhaps you manage a cloud-enabled technology and want to make your application cloud-native. Maybe you are planning a new software project and are wondering what architecture to choose. You want to start with cloud-native but you don’t know where to begin. Wherever you are, it is helpful to frame the state of cloud-native technologies so that you may make an informed decision on how to proceed.

Gartner[1] predicts that cloud-native platforms will be the foundation for more than 95% of new digital initiatives by 2025 - up from less than 40% in 2021. The State of Application Modernization Report 2022 by Konveyor[7] suggests that all businesses plan for modernization. The report surveyed 600 IT decision makers from the US, UK, and APAC. All respondents intended to modernize their systems. The difference in responses was how long they would take to modernize. Modernization can happen at your own pace, and the timing is not uniform across businesses.

Companies at the bleeding edge already experience many benefits of cloud-native technologies. Some companies still rely on traditional systems, and others use a hybrid approach as they transition to the cloud. Google's 2021 Accelerate State of DevOps Report[18] indicates that cloud adoption is in flux. 21% of companies are still on-premise, 34% use hybrid cloud, 29% use a private cloud, and 56% use public cloud. You will find that companies are at various stages of the transition journey, but the consensus is companies are thinking about when and how they should migrate, not if they should.

## Common barriers to the cloud-native journey

We know that moving towards cloud-native can bring many benefits for businesses. Before proceeding, companies often experience common barriers that prevent them from getting started. O'Reilly[11], Hashicorp[12], IDG[13], and Flexera[14] surveyed the most common barriers to modernization. For example, all surveys listed cost concerns, security, governance/compliance and lack of skills as barriers. Only O'Reilly and Hashicorp listed vendor lock-in as a challenge.



### Cost concerns

Shifting to the cloud has the potential to save on costs, but if left unchecked, the prices of platform-as-a-service can have no limit. The risk of uncontrolled cloud costs is a valid concern, but you can mitigate it through business rules such as cost limits and cost calculators. As discussed, cost savings is one of the main benefits of cloud-native technologies. When comparing a large-scale cloud-native system to an equivalent traditional system, the cloud-native system will be cheaper to run because of the economies of scale that the cloud provides. A study on DropBox suggests that highly specialized technology companies may benefit from running a custom storage infrastructure. Still, for the majority of users, cloud computing will be the path to cost savings.

The cost-saving concern may indicate the presence of the 'unknown unknown'. Companies know they should shift, but the costs are unknown, so cloud-native is a cost concern. By accurately estimating cloud costs, following the best practices for cost-optimization, and acquiring the necessary cloud skills to operate a cloud solution, the price may become a reason you would want to migrate to the cloud.

### Security

Moving to the cloud poses a security risk. Since data is online, data breaches can happen from anywhere in the world. The consequences of these breaches are devastating as companies may be carrying sensitive information. Securing access to cloud resources is a concern for companies. Security policies such as the 'need to know' need to be enforced to maintain security certifications such as the ISO-27001. Companies need to train employees on the correct data handling techniques. There may be a requirement for employees to complete a security training module. You are only as secure as your weakest link. If a portion of your company doesn't follow best practices, you can still experience the worst outcomes. Cloud technologies can provide you with all the right tools, but they can fail without the culture to execute the best practices. In fact, many data breaches are being caused by misconfigured clouds.

The primary risk of being "Internet-connected" is not new to most organizations. Cloud providers offer advantages in this area as they have adopted things such as ISO-27001 and have a larger budget for security than most organizations. One common data breach is storage accounts that are not private. For example, it's OK to store your website images on public storage accounts, but if people upload something containing PII to the same storage account, they are also making it public.

As with cost concerns, security is one of cloud computing's benefits. In a mature scenario following best practices, cloud computing follows security-compliant requirements. When companies try to implement cloud security without having the right set of skills, shortcuts are taken, which undermines the system's security.

#### Lack of skills/complexity

Some companies understand the argument for moving to cloud-native technologies but do not have the required skills. The benefits of cloud-native technologies, such as cost savings and security, are real, but they need specific skills to achieve them. Without these skills, these benefits become barriers and detract from the reasons to migrate. We list the most in-demand cloud computing skills in our first steps section.

One reason there may be a lack of skills is that cloud computing can be complex. There are several interdependent systems to manage, such as networking but also the use of hybrid cloud solutions. The complexity of these solutions presents a challenge to legacy businesses with minimal knowledge of the cloud.

#### Governance/compliance

Companies that migrate to the cloud must comply with relevant regulations for their country. The General Data Protection Regulation (GDPR) is a European Union (EU) regulation that protects the data of EU citizens. Any companies operating in the EU region can be prosecuted and fined for not complying with the law.

A bill of materials is a manufacturing term that lists the required inventory to produce a given output reliably. Bills of materials have been used for years to provide transparency and repeatability to the manufacturing process. Software bills of materials (SBOMs) apply a similar concept to bills of materials to software. SBOMs itemize the components in a software application in a list that developers can share across teams.

On the 12th of May 2021, The United States government released an executive order on Improving the Nation's Cybersecurity[15]. In the executive order:

> The Federal Government must bring to bear the full scope of its authorities and resources to protect and secure its computer systems, whether they are cloud-based, on-premises, or hybrid. The scope of protection and security must include systems that process data (information technology (IT)) and those that run the vital machinery that ensures our safety (operational technology (OT)).

The executive order seeks to minimize the cybersecurity risk in the supply chain that arises from acquired software. Cybersecurity risk increases as the number of unknown components in the software applications increases. The executive order requires all software bought by the US government to produce an SBOM. This has several implications for business inside and outside the US. Any US government project will now have a requirement to produce SBOMs for security purposes. Any vendor that cannot produce SBOMs for their products will not be approved to work with government projects. The executive order is likely the beginning of many similar orders to require SBOMs worldwide. As awareness of SBOMs increase, the more likely businesses anywhere will begin to demand SBOMs be made available.

Many companies comply with industry standards to meet audit requirements. ISO-27001 is just one of many types of certifications. There are also payment card industry rules and many others. These are often competitive requirements as you can't win contracts without these standards. Other legal requirements may depend on your organization's location or customers.

#### Culture/Vendor Lock-in

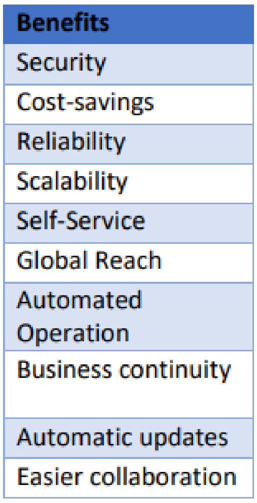
An organization's culture can affect cloud migration, such as preferences against a particular vendor. Culture would affect negotiations with cloud migration tools if a company does not want to be vendor-locked. Certain cultures may be risk-averse and prefer to use reliable legacy systems. Other cultures might be early adopters of every new technology. Despite best efforts to present a strong case for cloud migration, culture can get in the way.

According to the Puppet state of DevOps Report, companies that practice a DevOps culture perform well with cloud-native technologies. Cloud-native technologies are built on microservices and deliver agile, scalable applications. The quick release and development cycle of DevOps translates directly into developing cloud-native applications. DevOps methodologies allow for continuous monitoring and testing of your application. The Puppet state of DevOps Report has these insights:

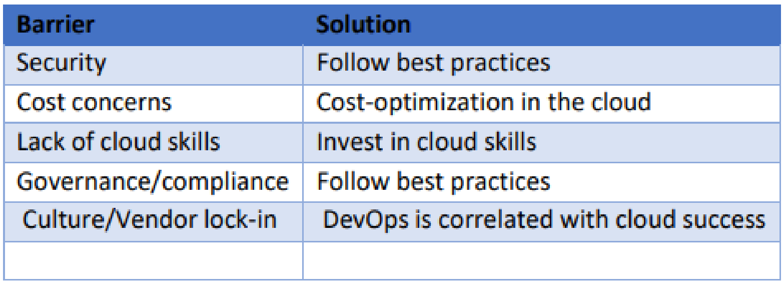
- Almost everyone uses the cloud, but most people use it poorly. However, highly evolved DevOps teams are using it well.
- Organizations should not expect to become highly evolved just because they use cloud and automation.
- While 2 in 3 respondents report using the public cloud, only 1 in 4 are using the cloud to its full potential.
- 65% of mid-evolution organizations report using the public cloud, yet only 20% of them are using the cloud to its full potential.

## Mitigating the barriers and realizing the benefits

When comparing the benefits of cloud computing to the barriers to entry, many of the obstacles can be mitigated by following best cloud practices in areas like security, cost, governance, and DevOps. The first table shows the benefits of migrating to the cloud. The second table lists the common barriers to migrating to the cloud. Each of these barriers can be mitigated by the suggested solution.



| Benefits |
|---|
| Security |
| Cost-savings |
| Reliability |
| Scalability |
| Self-Service |
| Global Reach |
| Automated Operation |
| Business continuity |
| Automatic updates |
| Easier collaboration |



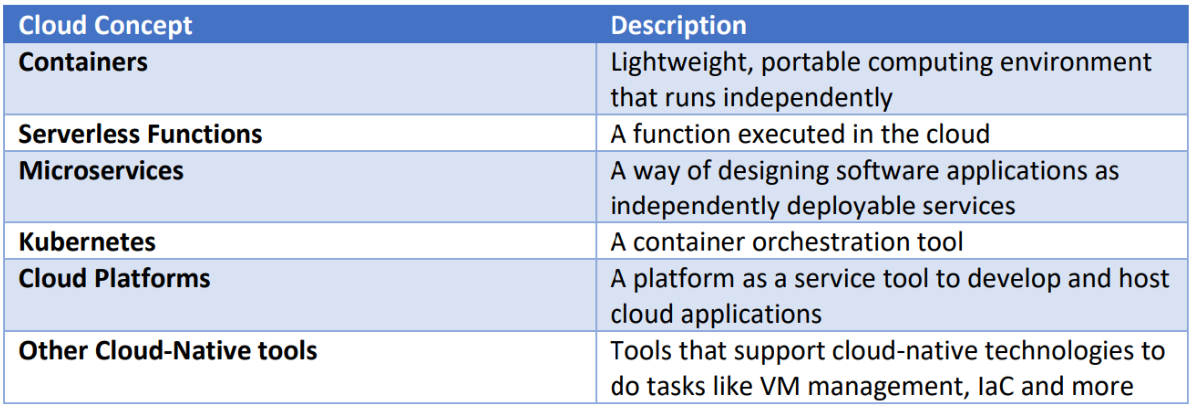
| Barrier | Solution |
|---|---|
| Security | Follow best practices |
| Cost concerns | Cost-optimization in the cloud |
| Lack of cloud skills | Invest in cloud skills |
| Governance/compliance | Follow best practices |
| Culture/Vendor lock-in | DevOps is correlated with cloud success |
| | |

## Part 2: The journey to cloud-native: the main concepts and how to get started

We have discussed that cloud-native technologies can provide competitive advantages to businesses over traditional systems. It can be hard to know where to start if you want to migrate to the cloud. To help you, we list the main concepts in cloud native, cloud-native best practices, and some first steps to get you started towards cloud-native.

### Cloud-native concepts

When you are getting started with cloud computing, there are a few main cloud concepts that are useful to understand. If you are already familiar with them or just want a quick refresher, refer to the table below:



| Cloud Concept | Description |
|---|---|
| **Containers** | Lightweight, portable computing environment that runs independently |
| **Serverless Functions** | A function executed in the cloud |
| **Microservices** | A way of designing software applications as independently deployable services |
| **Kubernetes** | A container orchestration tool |
| **Cloud Platforms** | A platform as a service tool to develop and host cloud applications |
| **Other Cloud-Native tools** | Tools that support cloud-native technologies to do tasks like VM management, IaC and more |

#### Containers

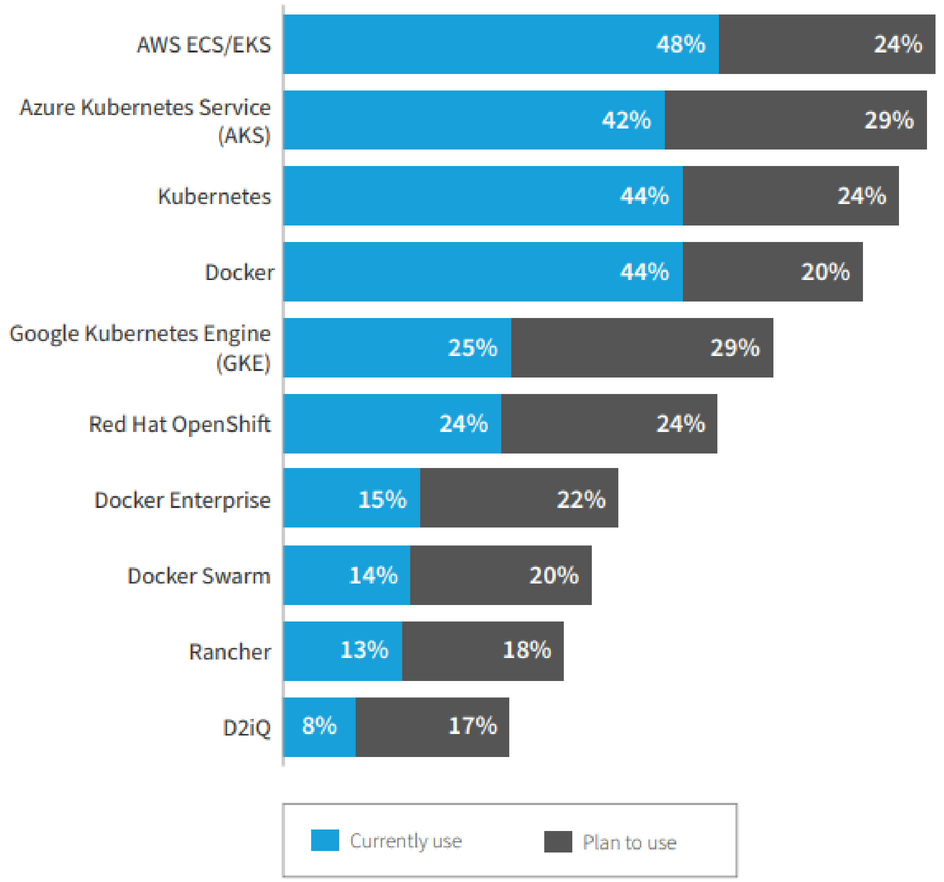
A container is a lightweight, portable computing environment with all the necessary files to run independently. Containerization is the process of making an application runnable as a container. Once the application can run as a container, it runs the same regardless of the infrastructure used to execute the container. Containers are loaded with container images that run a specific application inside the container. Containerization brings a level of consistency to the entire DevOps life-cycle. Because containers run consistently on a developer's local workstation or a cloud PaaS solution, DevOps teams can be confident that code will run as expected in production.

Containerization has been widely adopted in recent years, partly due to the availability of cloud technologies. Cloud technologies let you scale and replicate containers, lowering the barrier to entry. Since 2013, Docker has been the leading containerization technology. The three major cloud platforms (AWS, Azure, and GCP) have developed container-as-a-service offerings that complement the Docker ecosystem by providing custom environments to run containers. The report by Flexera highlights the widespread adoption of containerization through Docker or containers as service products.

**FIGURE 43**

Enterprises prefer to use *AWS ECS/EKS*, followed closely by *Azure Kubernetes Service (AKS)*, *Docker* and *Kubernetes*.

Enterprise use of container tools



### Serverless functions

Serverless functions are a cloud computing execution model where a function is written and stored in the cloud. When specific triggers occur, an instance of the function executes, and the server sends the result to the client. You can use serverless functions in web applications, the internet of things, or automation. You can convert any task in a function call into a serverless function.

Serverless functions remove the need for you to manage infrastructure. You can program serverless functions in all modern languages and execute them on different clients through APIs. Serverless functions run in the cloud and can be scaled up or down depending on demand. Serverless computing is not a magic bullet that can apply to any scenario. There are drawbacks to using it. A report by CloudFlare on serverless[19] lists the advantages of serverless:

- **No server management:** You do not have to deal with the servers as the vendor manages them.
- **Consumption-based pricing:** Serverless technologies employ a pay-as-you-go model where you only pay for what you use.
- **Scalability:** If a function requires more instances, serverless technology will add more instances automatically. Serverless will also destroy those instances when they are no longer needed.
- **Simplified deployment:** Since the vendor maintains servers, if you want to change a function, you can commit the code to the vendor, and the vendor will automatically apply the change across the servers.
- **Reduced Latency:** Vendors can run your function globally when you leverage the distributed network. The code runs closer to your end user, decreasing latency.

With the following negatives:

- **Challenging to test and debug:** Since you don't have insight into the back end, it is difficult to replicate the production environment if you want to test how a function may perform.
- **Security concerns:** If you are using the public cloud to host your functions, you may have to share infrastructure with another company. Sharing infrastructure may introduce security questions about multi-tenancy.
- **Long-running processes may not be cost-effective:** Serverless functions are most cost-efficient when short-running because of the pay-as-you-go model. Long-running functions may incur a high cost.
- **Warm-up performance impact:** Vendors charge a fee for starting up your process. If you frequently run the same function, you will have to manage what is known as a warm-start, where processes are kept ready for you on demand.
- **Vendor lock-in risk:** Depending on the vendor, a serverless architecture may expose you to vendor lock-in, where it is difficult to migrate to another vendor.

Serverless won't solve all cloud problems, but it can be another tool in the toolbox of software developers that they can use to deliver cloud-native technology.

#### Microservices

A monolith is a singular executable software application. It contains one codebase that defines the user interface, application logic, and database required to run an application. A monolithic application is built and maintained as a single, indivisible unit and is served from a central location. For example, in-house accounting software that uses a database to store customer records, a web server for record-keeping, and other components, all served from an on-premises server.

The term "Microservice Architecture" describes a way of designing software applications as collections of independently deployable services. Microservices are modularized. Microservices reduce the executed code's surface area, making it easier to test and update. Each microservice exposes an Application Programming Interface (API) or responds to other triggers like message queues to interact with other microservices.

APIs let microservices communicate with one another to execute a task. For example, a microservice sends input data to a second microservice. The second microservice sends input data to several other microservices, and so on. The same accounting software can run on the cloud, with the database, web server, and other components distributed across multiple geographical regions.

#### Kubernetes

If you are operating a cloud solution at scale, you will need an orchestration tool to orchestrate containerized workloads. Microservice applications leverage tools like Kubernetes to schedule and scale workloads. Kubernetes facilitates declarative configuration and automation where Kubernetes admins describe the cluster's desired state and Kubernetes reconfigures itself to match it.

According to the documentation[20], Kubernetes contains features such as:

- **Service discovery and load balancing:** Kubernetes locates containers via DNS or IP address. Kubernetes can load balance and distribute network traffic to stabilize the system
- **Storage orchestration:** You can automatically mount a storage system of your choice
- **Automated rollouts and rollbacks:** You can automatically apply a container update to all containers at once, and you can revert containers to their previous state
- **Automatic bin packing:** Kubernetes can optimize the resources for all of your containers, such as CPU and RAM. Optimization can be helpful when resources on your nodes are limited
- **Self-healing:** As mentioned, cloud-native technologies treat resources as expendable. Kubernetes will restart, replace or kill unhealthy containers to maintain uptime.
- **Secret and configuration management:** Kubernetes secures your environment by accepting OAuth tokens, passwords, and SSH keys in secrets. Kubernetes keeps your secrets in a configuration without

exposing them to your stack.

Kubernetes workloads are specified using YAML files, and we have developed a free tool that lets you construct valid Kubernetes YAML files[21] that will work in Octopus Deployments.

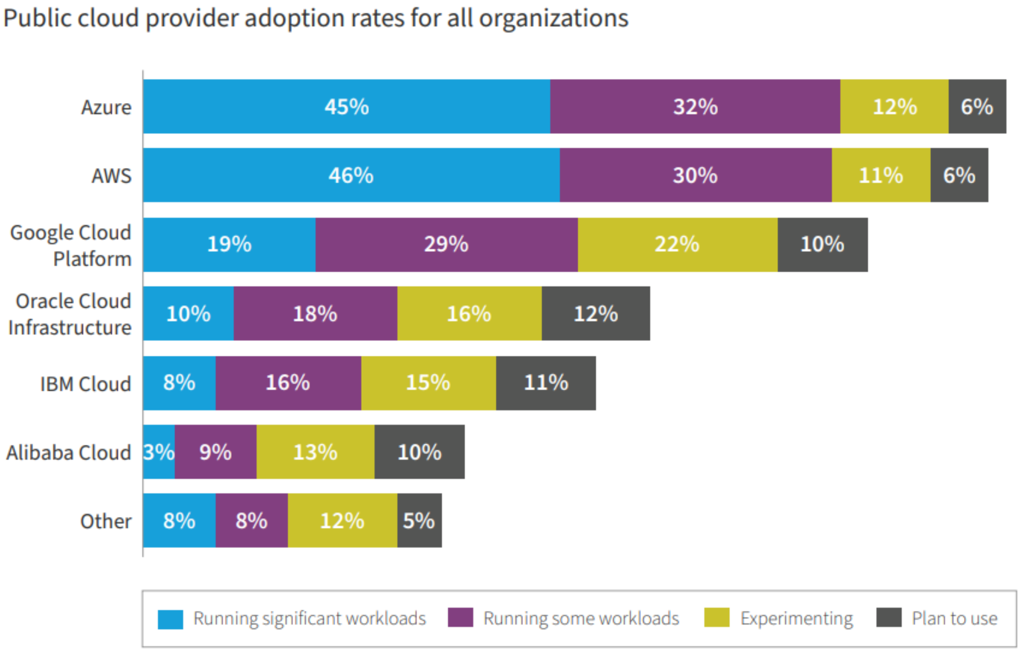
### Cloud-native platforms and tools

Cloud-native platforms allow users to develop and host applications in the cloud using cloud-native technology. Cloud platforms come with a comprehensive feature set. Starting from basic cloud-native units of works like containers and serverless functions to more niche applications such as AI or Blockchain, cloud platforms are self-service, and you select the mixture of tools you need to build your application.

Cloud platforms have a pay-as-you-go model where you only pay for the resources you use. You don't need to invest in going fully cloud-native from day 1. You can use hybrid solutions to maintain your traditional infrastructure and migrate your application to the cloud over time. The top cloud platforms today are Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP)

FIGURE 49

*Azure*, *AWS* and *Google Cloud Platform* lead the pack of public cloud providers.



According to the Gartner cloud management tooling list[22], some other popular vendors include:

- **Terraform:** An Infrastructure as Code (IaC) language that lets you represent and replicate your infrastructure with code.
- **VMWare:** Allows you to run Virtual Machines (VMs) on infrastructure.
- **Cisco:** Application resource management software.
- **Redhat:** Cloud management platform that handles virtualization application products.

Octopus Deploy is a top vendor for application release orchestration solutions[23], which falls under the cloud category. Octopus Deploy is a best-in-class continuous delivery tool, and we will discuss how we fit into cloud-native later on.

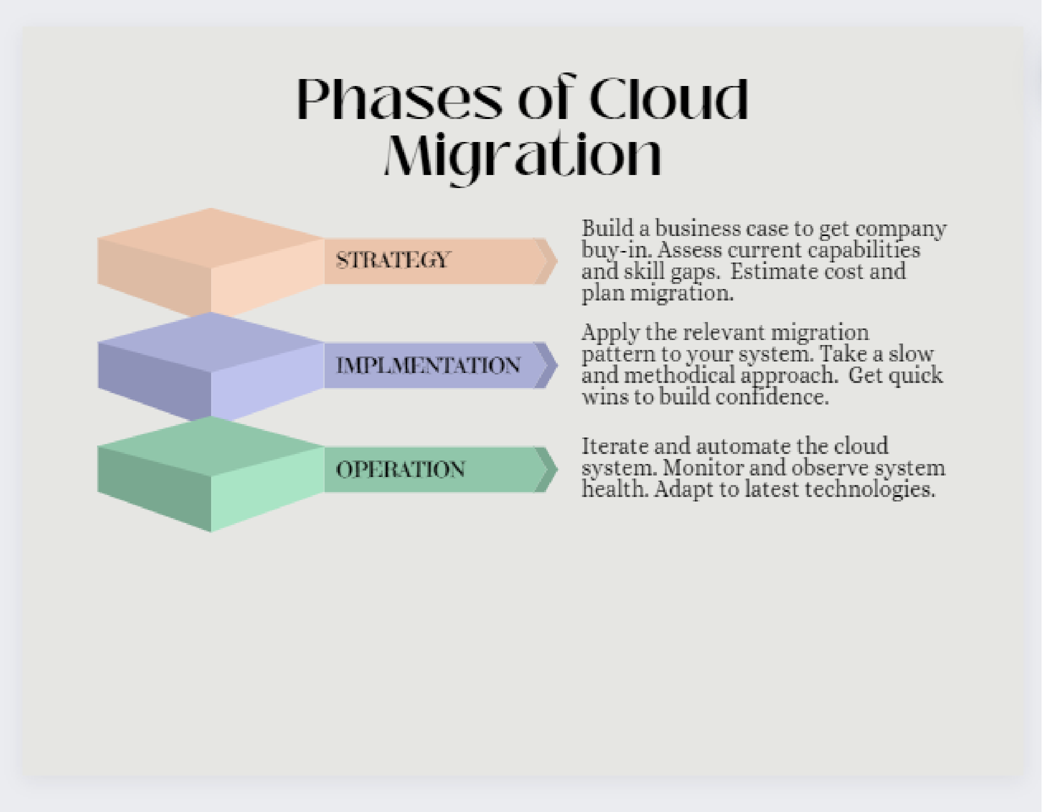
## Cloud Migration Best Practices: Strategy, Implementation, Operation

The migration to the cloud targets two types of people, cloud engineers, and cloud managers. Cloud engineers are concerned with implementation and operations, whereas cloud managers are concerned with planning and strategy. It is helpful to cover both of these concerns when migrating to the cloud. Businesses should migrate to the cloud at their own pace.

Companies can't stop and rewrite everything and must keep the businesses running while the migration happens. According to a study by ESG[24], organizations that adhere to migration best practices are:

- 3.7 times more likely to be completely satisfied with cloud services in use.
- 2.5 times more likely to report that typical savings for migrated workloads exceed 15 percent.
- 2.9 times more likely to report that they typically eliminated downtime for migrated workloads.
- 2.8 times more likely to report that the cloud has significantly improved their business.

The three leading cloud platforms, AWS[2], Microsoft[3], and Google[4], have all published reports on best practices to shift to the cloud. These reports share similarities, and the consensus is that there are three main areas that businesses have to focus on: strategy, implementation, and operation. Throughout the journey, it can help to have good partners such as Octopus Deploy that support you, whether you are on legacy, hybrid or modern.



Phases of Cloud Migration

STRATEGY

Build a business case to get company buy-in. Assess current capabilities and skill gaps. Estimate cost and plan migration.

IMPLMENTATION

Apply the relevant migration pattern to your system. Take a slow and methodical approach. Get quick wins to build confidence.

OPERATION

Iterate and automate the cloud system. Monitor and observe system health. Adapt to latest technologies.

A shift to the cloud requires strategic thinking that covers planning and assessing current capabilities. Once planning has been done, the next phase is implementing the plan across the business over time to shift towards cloud native. Once implemented, the new cloud functionality should be operated and monitored over time.

### Strategy

Before an organization can begin its journey towards cloud-native, there needs to be buy-in from senior management. Buy-in can happen in two ways, from the bottom-up or top-down. In a bottom-up scenario, internal teams adopt cloud technologies. It could start with a senior DevOps engineer that finds a DevOps tool that helps them automate their delivery pipelines(like Octopus Deploy). This engineer becomes an advocate for the software, and before long, multiple teams are using the tool and find it helpful for their workflows. This tool helps teams shift to cloud-native technologies, and eventually, there is a case to bring to senior management about moving the business towards the cloud.

In a top-down scenario, a senior manager may come across a report that lists the top 10 CRM SaaS tools and the reported improvements to productivity that come with using the tool. Using the testimonies of other people and the performance metrics cited, the senior manager advocates to other senior leaders for a company-wide adoption of a CRM tool.

In either a bottom-up or top-down scenario, senior management has to be convinced to shift to the cloud. To help with this, consider having clear, quantifiable goals that cloud-native solutions would provide. For example:

- a 50% reduction in the number of managed workloads in three months while spending 20% less on maintenance.
- 30% of company servers backed up in the cloud in 3 months.
- The automation of all continuous delivery pipelines with Octopus Deploy in Q1.

You could validate these goals through case studies run within the business. These goals should be defined and documented to present to management.

We believe a bottom-up approach to tool adoption is more sustainable than a top-down approach. There should be buy-in from not only senior management but also engineers that use the software on a day-to-day basis. We position ourselves as a software tool for software engineers and have added value by providing a best-in-class experience for engineers practicing DevOps.

Exploratory business analysis can assess existing infrastructure, workload forecasts, cloud networking needs, database requirements, and more. This analysis can be made more accessible by working with a cloud migration partner that can guide you on this. After an initial assessment, you can create a more detailed business case for modernization that includes cost estimates and timelines. Amazon suggests that if you can convince at least one senior leader, they can be your advocate from within the leadership team.

Cloud migration projects work best when there is accountability with someone passionate about moving to the cloud. This person would be a manager responsible for broader business goals and report cloud migration outcomes to senior management.

You may need to train or hire talent in your organization. AWS, Microsoft, and Google all offer cloud certifications to help develop internal skills. Cloud skills are transferable across different clouds, so your cloud practitioners can become skilled in multi-cloud. Multi-cloud skills can be advantageous if you want to change vendors in the future, and according to Hashicorp, 76% of companies are already multi-cloud.

Once the business plan has been presented and approved, you will build an end-to-end plan around timelines, milestones, resource planning, and funding. Any cloud migration project will be cross-functional, involving IT, Finance, and business owners. You will then meet with relevant stakeholders to implement the modernization plan.

#### Implementation

The implementation stage of cloud migration covers the patterns you will use to migrate your system. You will need a baseline level of cloud skills to execute these patterns. The skillset that we list is a good start. To migrate, you will need to choose a cloud platform. This choice will depend on your internal skillset and preferences. The major cloud platforms all have the essential tools required to start.

A migration pattern is a method you use to migrate a component in your system. There are many different patterns, each applicable in the right setting. According to Amazon[2], there are a set of common migration patterns known as the 6 Rs:

- **Rehost:** Also known as 'lift-and-shift', rehosting means moving an application and its data to the cloud without redesigning the application.
- **Re-platform:** Re-platforming is like rehosting, except that you must execute some cloud optimizations on top of the application before moving it. These optimizations do not change the architecture of the application.
- **Repurchase** Also called 'drop and shop,' involves replacing your current environment with a newer version, product, or solution.
- **Refactor:** Refactoring involves redesigning an application's architecture to align it with cloud-native requirements.
- **Retain:** This option is when you decide to keep your application on-premise and will consider migration to the cloud later.
- **Retire:** The retire pattern decommissions unnecessary components of your system.

Different migration patterns will apply to each component in your system. A hybrid approach is likely necessary as you move along the migration journey. Cloud-native does not mean that every company area must use cloud-native technologies. Even when you migrate, you should have the ability to run in the cloud or hybrid. Cloud-native should be 'freeing' rather than 'constraining'.

If you are beginning your migration journey and are unsure which pattern would apply to each component in your system, it is helpful to use a cloud migration partner. For example, if you wanted to migrate your deployments to the cloud, we could guide you on the best way to achieve this. Migration could mean optimizing the configuration of environments, tenants, and deployment targets to suit your needs. These concerns also apply to other components of your system, such as databases, security, networking, and more. Consultation with a cloud partner can set you up for success in each case.

The journey to cloud-native can happen at your own pace. You have to select which workloads to optimize, which to promote to the cloud, and which to maintain. Prioritization is a critical factor in migration because you may have to choose between competing components that need modernization. You may consider refactoring applications for productivity, rearchitecting applications for scalability in the cloud, or rebuilding them to be cloud-native. The path you choose depends on your budget and appetite for a temporary loss in productivity.

In the early stages of cloud migration, focus on quick wins that your team can achieve. These wins will build confidence and experience for larger projects. You will expand your scope as you migrate more components while iteratively modernizing your application. The mistakes you make and the lessons you learn while migrating components can help you update more effectively on future features.

#### Operation

Once you implement a cloud migration plan, you have to operate it. Monitoring is key to understanding how your cloud system responds to load. Through tools like telemetry, you can give your system observability. The metrics and traces from your system can give you a picture of system health so that you may take corrective actions if required.

You can manage security through cloud platforms through IAM roles, permissions, and access keys. A robust security capability is essential to compliance with regulations like GDPR or security certifications like ISO-27001. When you are operating a cloud, the consequences of non-compliance or a security breach are significant, so care has to be taken to manage the risk.

When you are in the cloud, you can take advantage of in-built modernization opportunities such as:

- **Infrastructure automation:** Once your infrastructure runs on the cloud, you will gain access to many automation features built into the cloud. IaC tools like Terraform will allow you to capture your entire infrastructure as a configuration file. You will be able to replicate your infrastructure on any compliant cloud platform.

- **Agile development practices:** An operational cloud solution will benefit from agile development practices. You will benefit from having a DevOps culture and agile development cycle as you modernize your system.

- **Cloud-native architectural patterns:** When you migrate to the cloud, you inherit the microservices architecture that powers cloud workloads. New cloud-native architectural patterns will emerge as you modernize and keep your application up to date. These could be blockchain-based, AI-based, or a new undiscovered pattern. Cloud technologies are at the cutting edge, and it is on the cloud platforms to deliver this capability for you to use and leverage.

- **Cloud-based momentum:** As you gain momentum in your cloud practices, you will build up a cloud capability for your system and team that will attract more talent. Your momentum will help further your company's expertise in the cloud and your cloud solution.

A cloud system will have running costs that you will manage. Cloud providers will let you limit your spending within a certain period. You may find more cost optimizations by selecting the appropriate cloud resources for the job. For example, cold line storage, a data bucket meant for infrequent access, costs much less than standard data buckets. You could save money by identifying data that only needs to be accessed a few times a year.

Cloud systems are complex, and automation can help you simplify the management of these resources. Orchestration tools like Kubernetes (usually paired with technologies like Fargate in AWS, Autopilot in GCE) can scale and shrink resource usage as needed. Orchestration tools unlock cost optimizations for your company. Cost optimizations can significantly impact the business case to continue your migration projects.

You can install observability tools on your system without changing your application code. These tools leverage the open-source OpenTelemetry API and build intelligent features. These tools include machine learning to help with intelligent alerts or generate actionable insights to improve the system.

If you're operating a modern cloud application, scanning for new tools and technologies to adopt continuously is helpful. We believe using the best tool for the job is essential because they help you deliver on best practices. Octopus Deploy is a best-in-class continuous delivery tool built to help you manage your deployments effectively.

## Cloud Migration Best Practices: First Steps

If you are starting your modernization journey, you may wonder where to start. You can make your first steps towards modernization by acquiring the relevant cloud skills, identifying and using the main cloud-native tools, and getting some quick wins to build momentum.

### The main skills required for cloud migration

A lack of skills was one of the most common barriers to migrating to the cloud. You should evaluate the talent you have in your organization and make sure that you have the right set of skills in-house before transitioning to the cloud. A skills shortage could mean hiring or retraining. According to O'Reilly[6], the primary skills required for the migration of cloud-based infrastructure are

- General cloud knowledge
- Containers

- Kubernetes
- Microservices architecture
- Cloud-based security
- Cloud-based regulatory compliance
- Monitoring
- Observability
- Scaling
- Performance tuning

Many of these skills are cloud engineering skills. In addition to hiring cloud engineers to fill these skills, hires for other roles such as software, DevOps, or system engineers would benefit from having these skills. The 2022 Open Source Jobs report by the Linux Foundation states that 77% of organizations are seeing growth in the use of cloud technologies and that there is growth in demand for cloud-native skills. All major cloud providers support cloud certifications that employees can acquire. These certifications test for knowledge of these essential cloud skills.

#### Main tools required to shift to cloud-native

If you want to shift to the cloud, you will need a base level of tooling to get started. The following list of essential cloud tools we think will help you move your business to the cloud:

- **Code Repository:** To store your application code, you need a code repository like Git to keep your code. Through Git functionality, multiple people can contribute to the codebase without conflicts, and you will always have a history of code changes.

- **Continuous Integration (CI) Server:** A CI server is a tool that lets you build your code and package it into a software artifact for deployment. Jenkins is the most popular CI server, and it is free and open source. You can automate your CI server to build your code on every commit through features like pipelines.

- **Continuous Delivery (CD) tool:** A CD tool helps you manage the deployment of your code. A CD tool takes the software artifact published by a CI server and deploys it to an environment such as Development, Test, or Production. The CD tool will deploy the artifact to deployment targets in the cloud or on-premise, where deployment targets will host the application.

- **Cloud platform:** If you want to migrate to the cloud, you will need to become familiar with a cloud platform. Your company may have preferences with dealing with a particular vendor, which will guide your decision. All cloud platforms have a trial period which you can use to assess the fit for your company.

- **DevOps Methodology:** To develop and maintain cloud-native applications, you should use a DevOps methodology in your developmental cycle. DevOps allows continuous improvement and feedback, which are essential to cloud-native technologies.

- **Workload Orchestrator** As you run workloads in the cloud, a workload orchestration tool helps to manage the dynamic workload. Tools like Kubernetes are essential to allow your application to scale. The major cloud providers also supply their workload orchestration tools like AWS ECS or AKS.

- **Infrastructure as Code (IaC):** IaC represents your system's infrastructure as a configuration file. This configuration file can be treated like code and stored in version control. The power of IaC lets you capture and replicate your entire cloud infrastructure in a single command. Hashicorp's Terraform is a popular IaC language that can help you achieve this.

- **Configuration as Code (CaC):** Like IaC, CaC stores all your configuration as code in version control. CaC allows you to revert to old configurations quickly and also make slight changes to configurations without fear of losing anything. At Octopus, we developed CaC as a first-class feature, allowing you to save your deployments in code.

#### Getting quick wins to build momentum

When migrating to the cloud, it is easy to fall into 'choice-paralysis'. There are many vendors to choose from, many components to upgrade, and many ways to do so. With so many choices, you can be paralyzed into not choosing anything. Amazon's case study on how to go fast[25] outlines some steps you can take to make a start and build momentum:

- **Just start:** At the beginning of a project, just get started instead of spending your time in the planning phase. Create a trial with a cloud vendor, create a trial with us, and just play around with what you can do. You will better understand what you want and don't want once you encounter the pain points rather than theorizing over them.
- **Maintain your purpose:** Understanding what you are trying to achieve is very important when you migrate. A migration effort will involve multiple meetings with cross-functional teams. Without a clear purpose throughout the migration project, different teams will work in isolation to meet their individual goals. Having accountable leaders that steer meetings and teams will help you stay on track.
- **Prioritize:** Prioritization is key to winning the migration battle. When you start a new project, there will be a thousand and one things you could work on, but not all are equally beneficial. Understanding which tasks do not matter and which are the most important can allow you to do less busy work, but more deliberate important work.
- **Don't reinvent the wheel:** When creating new cloud features and services, you don't want to reinvent the wheel. You want to be able to reuse the components you make in later projects. Designing your software for reuse supports the microservices architecture. Software reuse also applies to using existing features rather than creating your own. If there is a CI server or CD tool like Octopus Deploy, why not use that and benefit from prior learning instead of developing your own?

A project often gets stuck if there is no momentum. Once a project has momentum, it gets easier to see how it can succeed. You want to get a project to sufficient momentum as quickly as possible. By just getting started, moving along initiatives with purpose, prioritizing important work, and designing components for reuse, you can gain momentum for your cloud-based projects. Your team will see the wins and gain confidence and experience, which can help you deliver quicker on future projects.

## Part 3: Cloud-native and Octopus Deploy: a case study and our recommendations

Continuous delivery is the main component of cloud-native systems. You will need to select and use a CD tool like Octopus to deliver a modern application. Our tool works with legacy and modern systems so that you can keep your deployments while you are transitioning to cloud-native.

We also support hybrid solutions with the ability to place tentacles on legacy VMs and deploy to on-premises platforms like IIS, Tomcat, NGINX. We deploy to cloud platforms like Azure web apps, K8s, and ECS.

Our tool is entirely self-service, so if you can, you can deliver your deployment pipelines without any input from us. If you have a complex scenario, we have a support team that can assess your system and optimize your deployments to deliver your goals.

Your modernization scenario could be automating your legacy application CI/CD pipelines and strategically modernizing the ones that make sense for your company. Here are some of the ways you can use Octopus for cloud-native:

- **Configuration as Code:** We have a CaC feature that allows users to leverage the power of Git in their deployments. Users can create branches for new deployment processes and test the changes out in a separate release branch. Users can roll back to previous deployment configurations
- **Serverless:** Octopus provides serverless deployments for AWS Lambda
- **Cloud-native support for cloud platforms:** Our cloud-native support leverages our built-in cloud provider account support for AWS, Azure, and Google Cloud. Add your cloud provider credentials and securely use them throughout your deployments.
- **Automated cloud target discovery:** Octopus periodically checks if cloud infrastructure is available and removes it from the Octopus infrastructure if needed. This keeps your infrastructure dashboard clean and up-to-date.
- **Kubernetes support:** We support Kubernetes deployments and users can use either YAML or a UI experience to configure workloads. Our Kubernetes support allows for shared variables and deployment patters like blue/green, canary and rolling deployments.

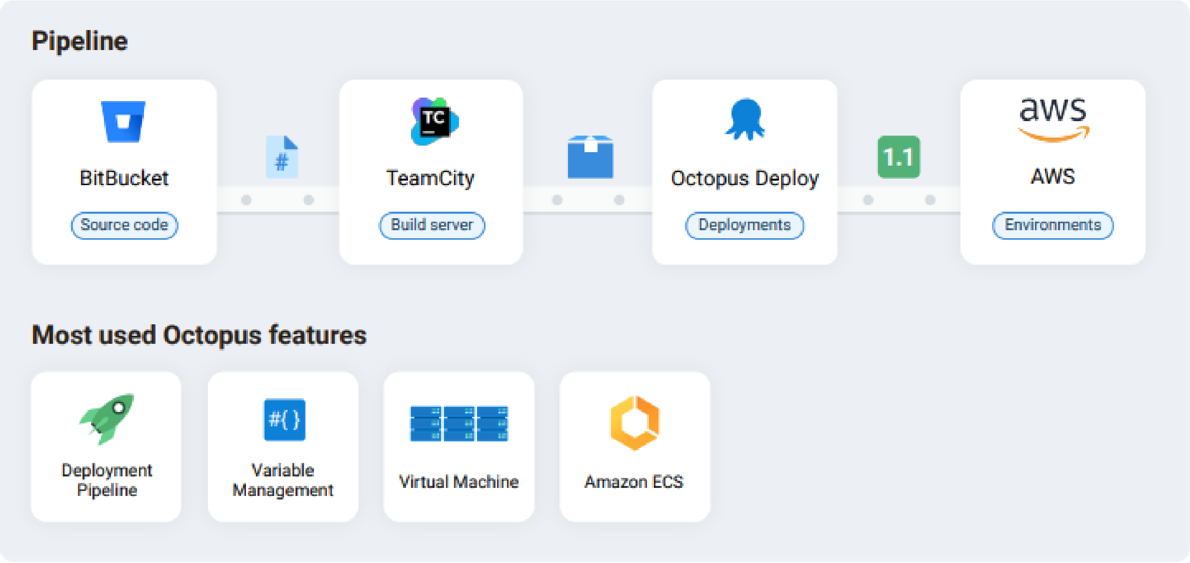
## Case study: EB Games

Our EB games case study demonstrates how modernization can help a business achieve greater success in the cloud. Prior to discovering Octopus Deploy, the EB Games team manually deployed their web applications by copying files and updating configuration files directly. EB Games has used Octopus Deploy for over 8 years and moved from manual processes to Octopus deploying to Windows Virtual Machines as they started to modernize their processes.

They ran this process happily for many years, primarily using Microsoft technologies to develop and host applications. They built their existing eCommerce platform using Microsoft ASP.NET and over time slowly began the process of teasing apart their monolithic application into smaller microservices and deployed it to Windows Server virtual machines running in AWS to host their websites and databases.

The team then looked to continue their modernization by containerizing some of the supporting eCommerce services and hosting them in Amazon ECS as they preferred the simplicity of Amazon ECS over Kubernetes. Their team manually configured their clusters with CloudFormation templates, and handcrafted their task definitions and custom scripts to automate their deployments and some maintenance tasks. This approach worked but was time consuming and required ongoing maintenance to maintain the custom scripts. It was also periodically prone to human error.

EB Games recognized the need for a better approach to enable better software delivery for their business. The next step of this modernization journey was shifting to Fargate backed Amazon ECS to further reduce manual handling in their deployment process.



Now, EB Games uses a best-in-class CI/CD pipeline that suits their team and needs. They use Atlassian BitBucket for source control, Jetbrains TeamCity for builds and Octopus for deployments to AWS.

Using Octopus Deploy to automate their CI/CD pipeline has been a game changer for the EB Games team. Without Octopus Deploy they would need to use custom scripting and custom configuration. The addition of Octopus has automated their pipeline and made deployments simple and easy to complete. The EB Games team doesn't have a dedicated build or deployment expert. The responsibility is shared among the team as the process is easy to learn and maintain.

Octopus continues to support their journey from legacy to modern by helping them shift from deploying their applications to Windows Virtual Machines to deploying to modern cloud native infrastructure with Amazon ECS. By supporting the right mix of infrastructure, Octopus is able to meet their team's automation needs over time.

The greatest benefit of using Octopus for the EB Games team has been the time saved from manually creating custom scripts and manual configuration. What should have been simple and straightforward was time intensive and required a lot of trial and error. Taking advantage of built-in automation steps and centralizing their configuration variables and secrets management in Octopus has solved this problem.

Based off the EB Games case study, we see a common pattern:

1. Get Octopus Deploy in place to deploy to your existing infrastructure.
2. Use a lifecycle to also deploy the application to an experimental cloud infrastructure (i.e. IaaS like a virtual machine), for example by having a pre-live "Cloud Test" environment.
3. After evaluation, go live with the application on cloud infrastructure / migrate data to the cloud.
4. Use Octopus to experiment with different clouds and cloud offerings using the "Cloud Test" environment, for example trying Amazon EKS vs Azure AKS to see what works best for the organization.

Octopus remains constant, even as you switch from IaaS to PaaS or serverless - which means it's easier to try a new offering. Being constant is where Octopus provides an advantage over tools you have to throw out if you switch vendors.

## Lessons from our journey to cloud-native

Cloud-native is a matter of business survival. We recognized that to be competitive, we needed to be cloud-native. Most businesses that look at the data will realize this as well. We started as an on-premise product but now have a fully-featured cloud-based offering. Our support for cloud-native technologies has come a long way.

We support cloud best practices and continuously improve our support for cloud-native technologies. We believe in cloud-native, so we focus on supporting cloud-native deployments. If you are looking at developing your cloud-native technology or updating your product to keep the latest cloud-native services, you may experience similar challenges as we did. Our recommendations are:

- Deliberate planning upfront will save time and clarify what you should be developing.
- Take it slow. You don't have to migrate to the cloud overnight. Be strategic in what you migrate and when. It is OK to have a hybrid solution, and we support it here at Octopus.
- Learn from your mistakes. As you migrate components, you will make mistakes and learn lessons you can apply to future migration projects. Capture your learning in documents or logs that you can refer back to later.
- Involving your customers and getting early feedback is crucial to developing a product that fits the needs of your user base.
- Find a good cloud partner when migrating, learn from the best, and use the best tools for the job.

- With any cloud-native product, your support for it should be first-class, not a secondary concern. Users expect their best user experience, so don't let your product be the one that doesn't live up to their best knowledge. We made sure that our Configuration as Code feature delivered a best-in-class experience.[36]
- Get your feet wet by getting started. Most cloud providers have a free trial that you can use to try the product out. Why not start a free trial with us and see what we can do for your deployment process?

## Conclusion

The cloud provides always-on, scalable and reliable services that can bring many benefits to a business. Shifting to the cloud is a challenge that most businesses are faced with. Modernization is a journey that takes planning and time. Every company is at different stages of modernization and it can happen at your own pace.

There are three primary phases to modernization: strategy, implementation, and operation. To get started, you need cloud skills, the main cloud tools and to get quick wins to build momentum. Our first steps to gaining momentum in your cloud projects are:

- **Just start:** At the beginning of a project it is important to just start by creating trials with cloud software to get hands on experience with what you can and can't do.
- **Maintain your purpose:** Aligning your cross-functional teams across a central purpose will help you work towards your cloud goals.
- **Prioritize:** Understanding which tasks are the most important can allow you to do less busy work, but more deliberate important work.
- **Don't reinvent the wheel** Reusing existing components in new projects or using libraries to accomplish common tasks can streamline your cloud projects.

We believe that modernization is achievable for most companies. Even if your pace towards cloud-migration is slower than more agile companies, that is still OK. Not all of your components need to be cloud-native immediately, there are strategic ways to migrate your systems by utilizing a hybrid approach. You will be able to see some tangible results be consistently applying the migration tips in this white paper.

From our director of product, Michael Richardson:

> Shifting to cloud-native can be difficult, particularly at scale. It isn't easy to lift-and-shift every component, and technologies such as Kubernetes and ECS can be complicated with plenty of configuration issues. Modernizing your CI/CD pipeline is about combining the right set of technologies and tools, and choosing a strategy that is flexible to where you need to modernize and maintain. No matter where you are on your cloud journey, Octopus is the only deployment tool you will ever need. Whether you are deploying to AWS, Azure, GCP, or your own data-center, Octopus helps you modernize to containers and cloud-native with flexibility as you transition from getting started to full control for more complicated deployments.

If you would like to get started, you can:

- sign up for a free trial
- sign up for the newsletter
- view our free tools
- upgrade to the latest version

## References

1. https://www.gartner.com/en/newsroom/press-releases/2021-10-18-gartner-identifies-the-top-strategic-technology-trends-for-2022

2. https://pages.awscloud.com/rs/112-TZM-766/images/AWS_Migration_8_Best_Practices_ebook_final.pdf
3. https://azure.microsoft.com/en-au/migration/migration-journey/#how-to-migrate
4. https://cloud.google.com/architecture/migration-to-gcp-getting-started
5. https://docs.microsoft.com/en-us/dotnet/architecture/cloud-native/definition
6. https://12factor.net/
7. https://www.konveyor.io/modernization-report/?utm_source=thenewstack&utm_medium=website&utm_campaign=platform
8. https://octopus.com/blog/benefits-of-containerization
9. https://octopus.com/blog/monoliths-vs-microservices
10. https://aws.amazon.com/solutions/case-studies/brightline-case-study/
11. https://www.oreilly.com/radar/cloud-adoption-in-2020/
12. https://www.hashicorp.com/state-of-the-cloud
13. https://www.dropbox.com/s/y563i0nq1gmw6b8/2020%20Cloud%20Computing%20executive%20summary_v2.pdf?dl=0
14. https://resources.flexera.com/web/pdf/Flexera-State-of-the-Cloud-Report-2022.pdf
15. https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/
16. https://www.dropbox.com/s/nnagymzdcnoswc6/Benchmark-Report-State-of-Testing-in-DevOps.pdf?dl=0
17. https://www.dropbox.com/s/xycst8qsxnpsieu/state-of-devops-2021.pdf?dl=0
18. https://services.google.com/fh/files/misc/state-of-devops-2021.pdf
19. https://www.cloudflare.com/en-au/learning/serverless/why-use-serverless/
20. https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/
21. https://k8syaml.com/
22. https://www.gartner.com/reviews/market/cloud-management-tooling
23. https://www.gartner.com/reviews/market/application-release-orchestration-solutions
24. https://pages.awscloud.com/rs/112-TZM-766/images/ESG%20AWS%20Cloud%20Migration%20Best%20Practices%20Study%20Final%20Aug2019.pdf
25. https://aws.amazon.com/blogs/enterprise-strategy/how-to-go-fast/
26. https://octopus.com/blog/kubernetes-containers-update#summary
27. https://octopus.com/blog/octopus-release-2018.8
28. https://octopus.com/blog/deploying-applications-to-kubernetes
29. https://github.com/OctopusDeploy/TenPillarsK8s/releases/tag/0.1.269-main
30. https://octopus.com/blog/ultimate-guide-to-k8s-microservice-deployments
31. https://octopus.com/blog/octopus-kubernetes-yaml-generator
32. https://octopus.com/blog/why-kubernetes-and-octopus-deploy
33. https://octopus.com/blog/ecs
34. https://octopus.com/blog/rfc-ecs-integration-with-octopus
35. https://octopus.com/blog/rfc-second-ecs-integration-with-octopus
36. https://thenewstack.io/our-approach-to-config-as-code-at-octopus-deploy/