Cloud Migration Checklist

Step 1: Assign a migration manager

What you thought you knew about cloud migrations, there's a good chance that you're wrong. Before you begin your cloud migration journey, whether you're migrating from an on-premises environment or transitioning vendors, you should consider consulting an expert in migrations.

AWS partners are experts and will be able to analyze your current technology stack and give you specific recommendations based on your goals and requirements. Partners will be able to help guide you through the steps that follow.

If you have somebody in your organization with a proven track record of migrating workloads to the cloud, that's your second-best option.

Whoever that person ends up being, assign them as the migration manager to oversee and lead the migration. The migration manager is a system architect-level position responsible for planning and completing all aspects of the migration.

Their core responsibility should include defining the necessary refactoring required to make the migration successful, designing strategies for data migration, defining cloud-solution requirements, and determining migration priorities and production switchover mechanisms.

During a large migration project, there are many decisions and technical plans that must be made, and having a migration architect who is responsible for all aspects of the migration is critical to the success of the project.

Step 2: Select Level of Cloud Integration

When you move an application from an on-premises data center to the cloud, there are two ways you can migrate your application—a shallow cloud integration or a deep cloud integration.

For a shallow cloud integration (sometimes called “lift-and-shift”), you move the on-premises application to the cloud, and make no—or limited—changes to the servers you instantiate in the cloud to run the application. Any application changes are just enough to get it to run in the new environment. You don’t use cloud-unique services. This model is also known as lift-and-shift because the application is lifted “as is” and moved, or shifted, to the cloud intact.

For a deep cloud integration, you modify your application during the migration process to take advantage of key cloud capabilities. This might be nothing more advanced than using auto-scaling and dynamic load balancing, or it might be as sophisticated as utilizing serverless computing capabilities such as AWS Lambda for portions of the application. It might also involve using a cloud-specific data store such as Amazon S3 or DynamoDB.

Step 3: Choose a single cloud or go multi-cloud

Before you begin your cloud migration, address this question: Do you want to pick a single cloud provider and migrate your application so it runs optimized for that single environment, or do you want your application to run on multiple cloud providers?

Optimizing your application to work with a specific cloud provider is relatively simple. Your development teams have just one set of cloud APIs to learn, and your application can take advantage of everything your chosen cloud provider offers.

The downside to this approach is vendor lock-in. Once you’ve updated your application to work with only that one provider, moving your application to a different provider could require just as much effort as the original cloud migration. Additionally, having a single cloud provider might negatively impact your ability to negotiate important terms—such as pricing and SLAs—with the cloud provider.

But wait, it gets even more complicated. There are several different models for using multiple cloud providers:

One application in one cloud; and another application in a different cloud. Perhaps the simplest multi-cloud approach runs one set of applications in one cloud provider and another set in another. This approach gives you increased business leverage with multiple

providers as well as flexibility for where to put applications in the future. It also lets you optimize each application for the provider on which it runs.

Split your application across multiple cloud providers. Some companies choose to run parts of an application in one cloud provider and other parts of it in another. This approach lets you utilize key advantages each provider offers (for example, one provider might have better AI capabilities than another, which is known for its database speeds). The risk here is that your application is tied to the performance of both providers, and any problems with either provider could impact your application's customer experience.

Build your application to be cloud agnostic. Other companies build their applications to run on any cloud provider. With this approach, you could run your application simultaneously on multiple providers or split your application load across them. This model gives you the ultimate flexibility in vendor negotiations because you can easily shift loads from one cloud provider to another. The downside is that you may find it difficult to use the key capabilities of each cloud provider, reducing the benefits of hosting your application in the cloud. This approach may also complicate your application-development and validation processes.

Step 4: Establish cloud KPIs

Key Performance Indicators (KPIs) are metrics that you gather about your application or service to measure how it is performing against your expectations. You may already have defined some KPIs for your applications and services, but are they still the right ones for an application or service once it's in the cloud? The best KPIs for a cloud migration show how your in-progress migration is doing, illuminating visible or invisible problems that may be lurking within your application. Most important, perhaps, cloud migration KPIs can help you determine when the migration is complete and successful.

There are several key categories of cloud migration KPIs:

Category	Sample KPI
User Experience	Page Load Time
	Lag

	Response Time
	Session Duration
Application/Component Performance	Error Rates
	Throughput
	Availability
	Apdex
Infrastructure	CPI Usage %
	Memory Usage
	Network Througput
Business Engagement	Cart adds
	Conversion and and Conversion %
	Engagement Rates

For each category, determine which metrics are the most important to your business, and which metrics will be most impacted by the migration to the cloud.

Step 5: Establish performance baselines

Baselining is the process of measuring the current (pre-migration) performance of your application or service in order to determine if its future (post-migration) performance is acceptable. Baselines help you determine when your migration is complete and provide validation of the post-migration performance improvements you expected. You can also refer to baselines during a cloud migration to diagnose any problems that arise.

Set a baseline metric for each KPI that you've decided to measure. Determine how long you will collect data to determine the baseline. Choosing a short baseline period (such as a day) lets you move faster, but you risk not collecting a representative performance sample. Choosing a longer period to baseline (such as a month) obviously takes more time, but can provide more representative data.

You also need to determine if you want to collect only baseline data that's average or representational, or if you want to include data collected over "peak" or "critical" periods. For instance, if you're a news site, do you want to collect data over a day with a big news event, or do you want to avoid such days?

No matter which data-collection model is appropriate for your industry, be sure to clearly define what type of data you're going to collect and for what period of time.

Step 6: Prioritize migration components

You also have to decide if you will migrate your entire application at once, or if you will migrate it to the cloud component by component or service by service.

First, identify the connections between your services, and which services depend on what other services. For larger, more complex applications, use an application performance monitoring tool that can use service maps to generate dependency diagrams. Use the dependency diagram to decide which components should be migrated and in what order. It often makes sense to start with the services that have the fewest dependencies. In this case, you'll migrate your most internal services first, and then follow up with your outermost services, typically the ones closest to your customers. The alternate approach is to start with the services closest to your customers—the most outside services—so that you can control any impact on your customers.

Step 7: Perform any necessary refactoring

You may want to do other work on your applications and services before you migrate them so they work as effectively and efficiently in the cloud as possible. For example, you may want to refactor your application:

So it works effectively with a variable number of running instances to allow dynamic scaling, potentially saving you money on cloud service costs.

So your resource utilization can better take advantage of dynamic-cloud capabilities, such as the ability to dynamically allocate and deallocate resources as needed, rather than you statically allocating them ahead of time.
To move to a more service-oriented architecture before the migration, so that you can more easily move individual services to the cloud.

Step 8: Create a data-migration plan

Migrating data is one of the trickiest parts of a cloud migration. The location of your data can significantly impact the performance of your application. Moving your data to the cloud when the data-access methods are still primarily on-premises can significantly impact performance. The same holds true if the data is still on-premises but the service accessing it resides in the cloud.

Options for data-migration include:

Using a bi-directional syncing mechanism between your on-premises and cloud databases. Once you've moved all consumers of the data to the cloud, remove the on-premises database.
Use an on-premises database with one-way synchronization to a cloud-based database, and allow consumers to connect only to the on-premises version. When you're ready, disable access to the on-premises version so the cloud-based version becomes the main database, and enable cloud-based consumers access to the new database.
Use a cloud data-migration service, such as those available from Amazon Web Services.
Don't underestimate the complexity and importance of data-migration planning. Not paying close attention to your data-migration plan before you begin a cloud migration can cause migrations to fail, or at least fail to meet expectations. Your migration architect should be very involved in the data-migration planning process.

Step 9: Switch over production

When and how do you switch over the production system from the legacy on-premises solution to the new cloud version? The answer depends on the complexity and architecture of your application, and especially the architecture of your data and datastores.

There are two common approaches:

Do it all at once. Wait until you've moved the entire application or service over to the cloud and validated that it works there, and then switch traffic from the on-premises stack to the cloud stack.
Do it a little bit at a time. Move a few customers over, test that things are still working, and then move a few more customers. Continue this process until you've moved all your customers to the cloud-based application.

Step 10: Review application resource allocation

Even after you've finished migrating everything to the cloud, there are a few more things to consider. Most important is resource optimization. The cloud is optimized for dynamic resource allocation, and when you allocate resources (servers, for example) statically, you're not taking advantage of the cloud's strengths. As you move into the cloud, make sure your teams have a plan for distributing resources to your application. When you need to allocate additional resources to an application in the cloud, they are usually available from the vendor in virtually any quantity in a moment's notice. This means that you can typically trust that you can scale as needed to meet demand, assuming your teams have the application architecture in place to support dynamic scaling.

Other considerations for your cloud migration

The 10 steps in this cloud migration checklist cover a lot of ground, but there are definitely other things you should consider during your cloud migration. Creating a safe and secure cloud environment, for example, is obviously a critical part of any cloud migration. Fortunately, the major cloud providers offer significant tooling and resources to help you build and maintain a secure system.

When it comes to cloud costs, there are two rules of thumb about cloud pricing: the cloud is cheaper than on-premises, and the cloud is more expensive than on-premises. Both can be true or false, depending on the situation.

It is certainly possible to start using the cloud and find out that your infrastructure bill has, in fact, increased compared to what you were spending on your physical data center. There are a couple reasons this could happen:

First, there are hidden costs in all infrastructure systems, and you may not be considering all the costs involved in running your own data center, while the monthly bill you get from your cloud provider makes the costs very clear. The result? Sometimes you end up comparing apples to oranges, making the cloud solution appear more expensive than the on-premises one.

Second, the costs of an on-premises infrastructure are mostly composed of capital expenditures (CapEx), while a cloud-based infrastructure usually comes out of operating expenses (OpEx). Depending on how your business manages its books, CapEx may be easier to come by than OpEx, or vice versa. Understanding how paying for cloud-based infrastructures differs from an on-premises infrastructure, and making sure your company's financial models support the distinctions, is critical to recognizing cloud cost improvements.

For more on the topic of cloud costs, check out How to Calculate the Cost of a Cloud Migration.

Finally, I recommend anyone looking at planning a cloud migration to familiarize themselves with topics like building modern applications using services and microservices, especially twelve-factor applications, and using DevOps methods and procedures as best practices for building and running cloud services and applications. Oh, and don't forget to optimize your customer experience once you're fully migrated to the cloud.