White paper

Four Paths to Consider for a Successful Migration

rackspace technology®

Customer First.
Cloud First.

Cloud migration is a top priority

According to the Flexera 2022 State of the Cloud Report, businesses are only running about 50% of their workloads in the cloud. It is expected that this will rise by 6% in just the next 12 months. Despite this, there still remains some confusion about the pathway options available when migrating to the cloud.

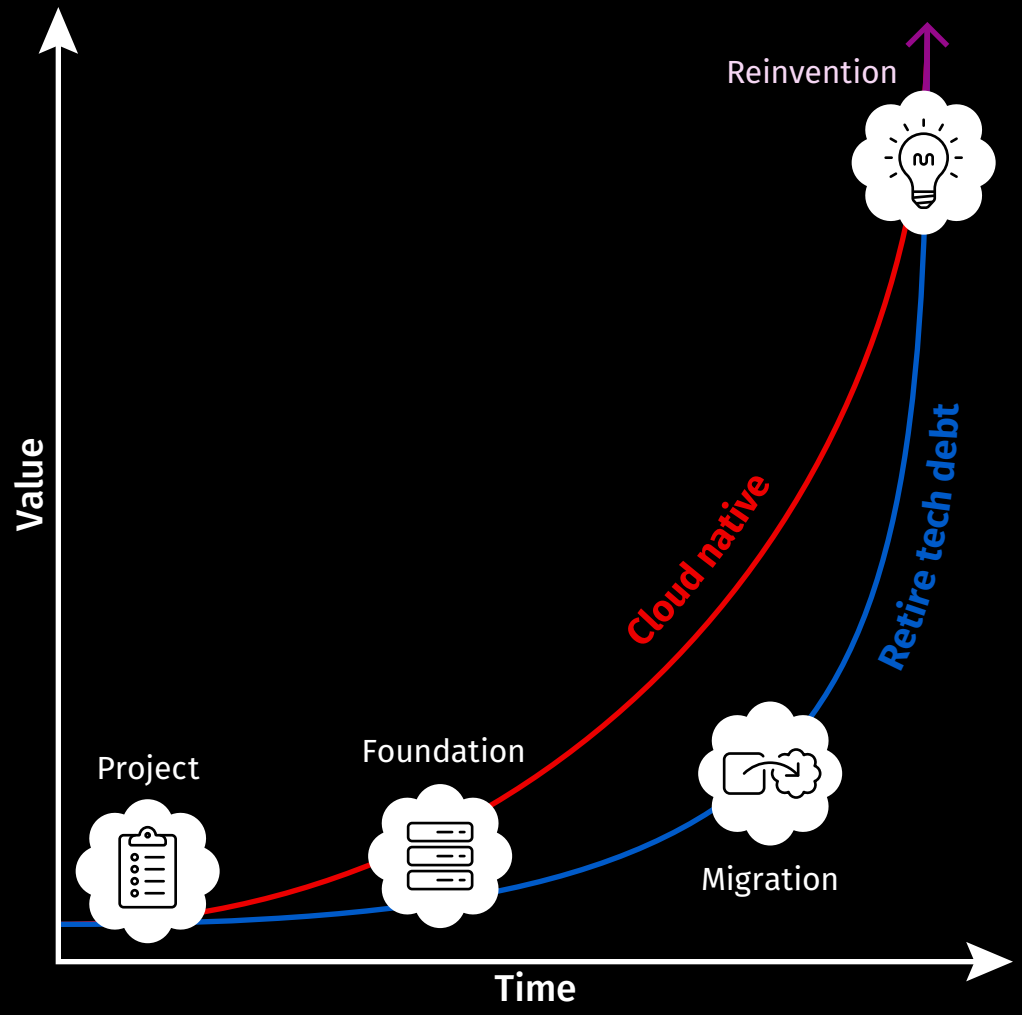
Deciding how to migrate should be a process that focuses on each individual workload, rather than trying to find a single solution to use across the board. Key to these decisions are factors around: the impact on the business and customers (market differentiation), level of engineering effort and resource, time constraints, licensing restrictions, and In-house versus commercial-off-the-shelf software.

As you create your migration plans, there are four pathways worth considering. This guide provides an overview of the paths — from most to least ideal:

1. **Refactor** for cloud native design
2. **Repurchase** to automate with cloud-native architecture
3. **Replatform** for partial automation
4. **Rehost**

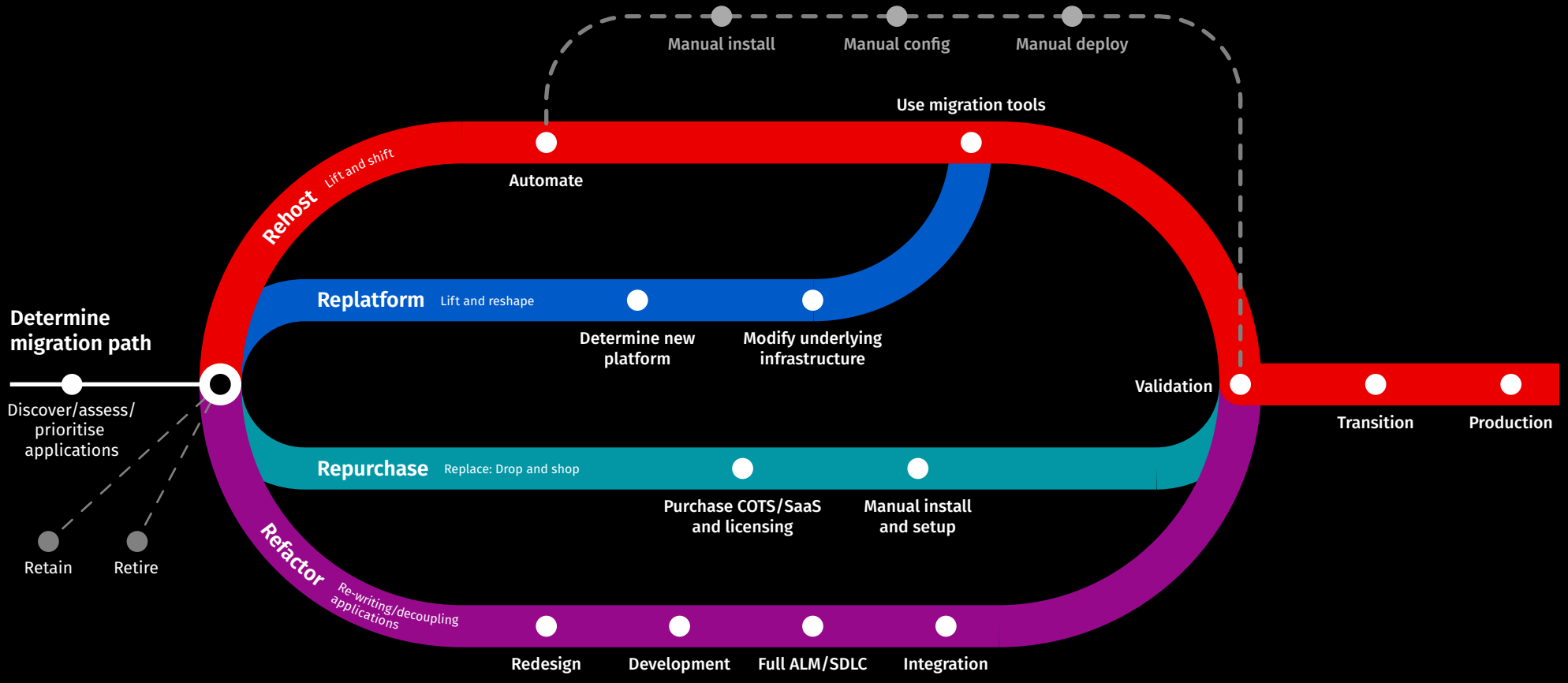
Customer First. Cloud First. For better outcomes.

Cloud-enabled digital transformation enhances your abilities to innovate, create new revenue streams, build better customer experiences and establish new models for work and collaboration. As you move through your migration journey, we help ensure that all of your technologies work together to move you toward these outcomes.



rackspace technology.

The four migration paths — and the steps in each journey



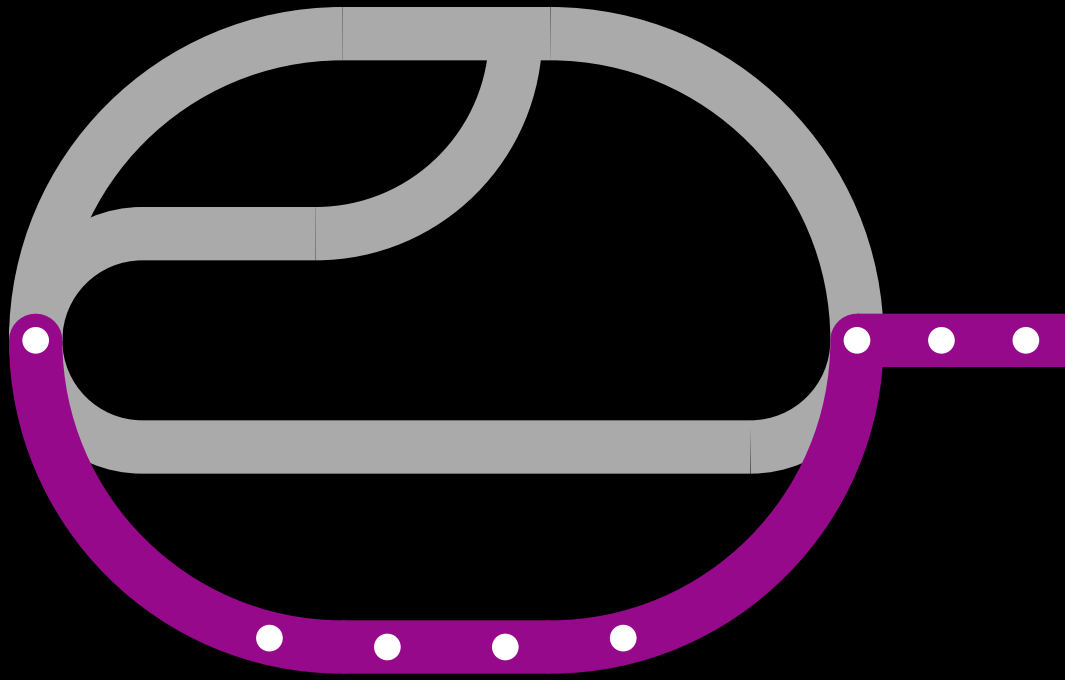
1. Refactor for cloud native design

Refactoring for cloud native is the migration path that requires the most forethought, planning, engineering effort and time to implement. However, the leading benefit is that it is the most stable path.

Refactoring existing applications as they are currently used and written allows for the option to fully automate the build/test/release cycle processes (leveraging more modern CI/CD pipelines), and ensures that old workloads get updated to modern processes and architectures. The opportunity to take an old, problematic application and re-write it to be cloud native may include serverless options, the opportunity to incorporate native backup and restore solutions, disaster recovery or highly distributed design and architecture, and anything else that would allow you to benefit from the services provided by the public cloud. Often this even includes moving away from existing data structures toward a more modern database engine, providing an opportunity to escape expensive licensing by choosing open-source standards and more.

Planning for the refactor migration path requires extensive research to identify and resolve automated processes around every aspect of the workflow. Concepts common to this model would require DNS automation, service discovery, blue-green or canary deployment, centralised logging, version control workflows, artifact creation, storage and more.

A high level of maturity is strongly recommended for making these decisions, whether that's internally or through the use of a trusted partner. This is not necessarily always an option, however. The major factors that play into this option are the obvious ones: time and money. An organisation may not have the engineering resources or maturity to implement a custom, cloud native solution to an off-the-shelf application currently in use today. Even in-house applications may not be worth refactoring within the given time, or other business priorities may make it an impossibility. Without available engineering resources, the company is left in a position to hire new talent with those skills, train their existing workforce and expect slower delivery of the solution, or simply choose an alternative path.



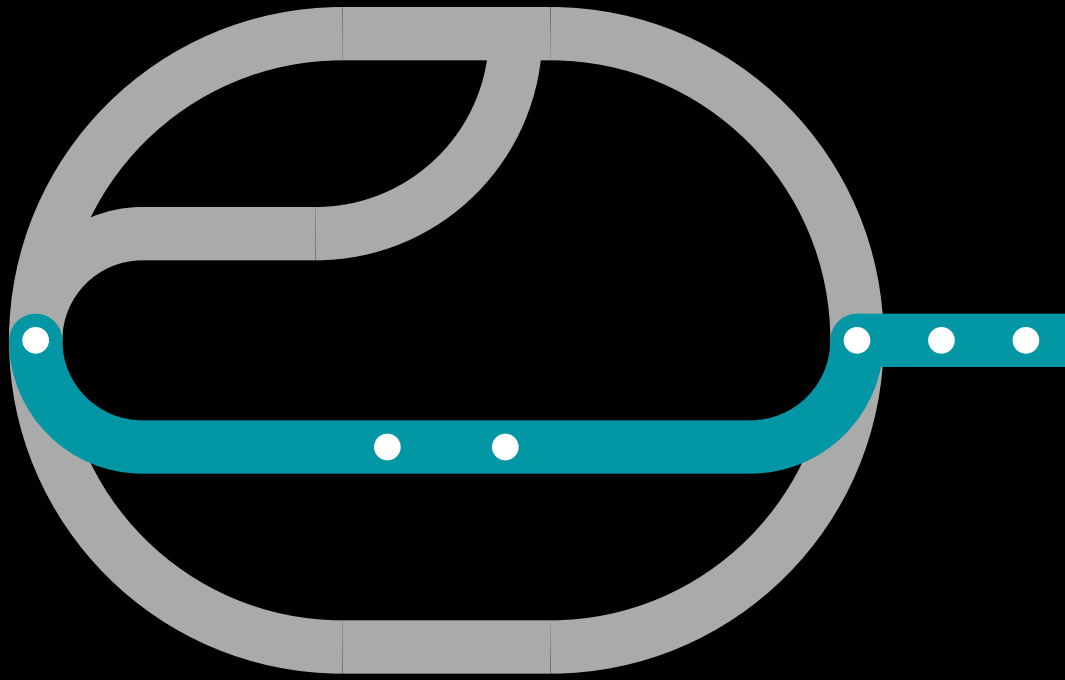
1. **Redesign application/infrastructure architecture**
2. **Application code development**
3. **Full ALM/SDlC**
4. **Integration**
5. **Validation**
6. **Transition**
7. **Production**

Arriving at a decision to redesign (modernise) an application can be complex. See the Rackspace Technology report on the State of Application Modernization. Also, see how Truecar refactored its core business and cut costs by 50%.

2. Replace with cloud native architecture

Straying a little further away from a full refactor is the idea of taking the same applications and mindsets, but migrating them with more cloud native architecture. This means using automation from start to finish, codifying all infrastructure and designing with the benefits of the public cloud in mind for availability and disaster recovery. An example of this approach would be to take a workload in your data centre and write functional code to deploy its infrastructure and application installation into the cloud in such a way that it's highly available (self-healing with autoscaling groups and effective health checks).

There are many of points to consider when replacing an application with software as a service (SaaS). For more details, check out our white paper, To Build or To Buy?



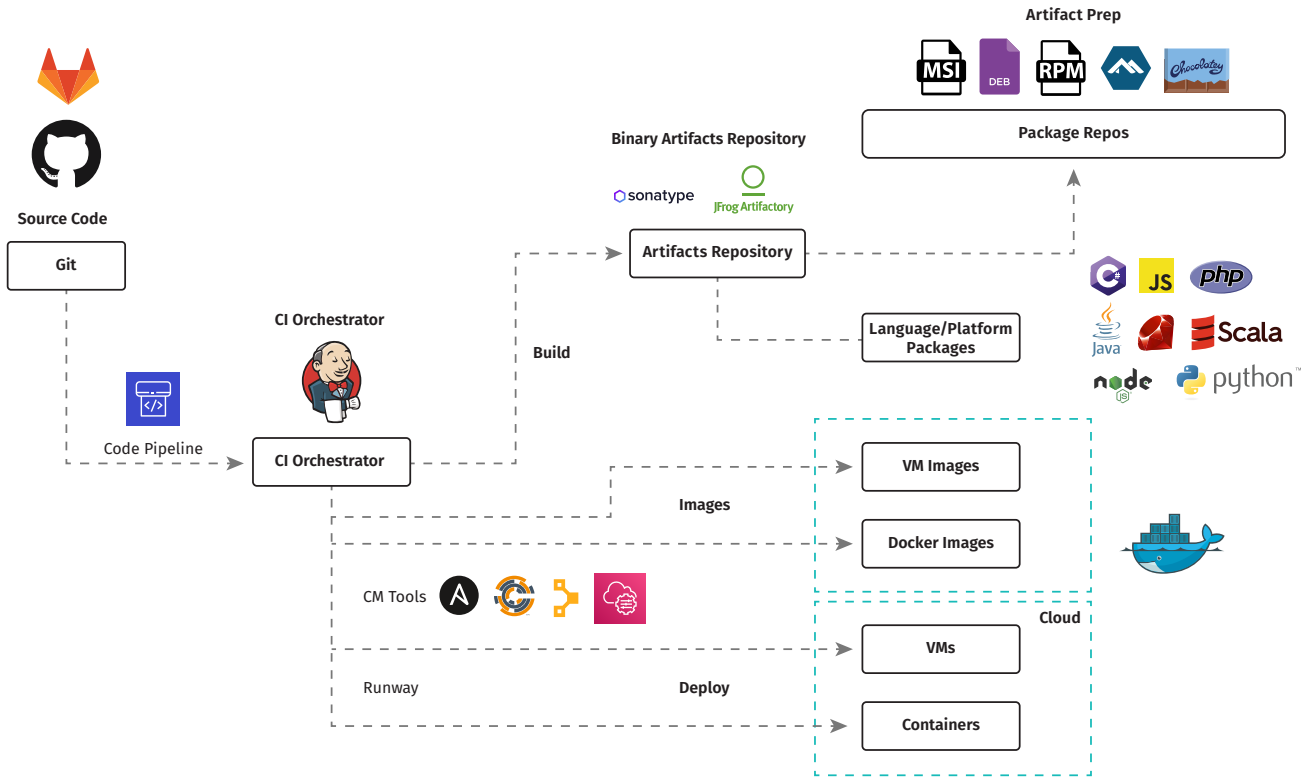
1. **Purchase COTS/SaaS and licensing**
2. **Manual install and setup**
3. **Validate**
4. **Transition**
5. **Production**

An overview of pipeline-driven migration

Many, if not all, of the external processes that would need to be identified for repurchase and replace pathway are the same as a full refactor, as things like DNS and service discovery may prove to be necessary components for the levels of automation that are intended. Deployment processes need to be identified, artifacts still need to be built, and code needs to be written for all of these things to talk to each other. The Migration as Code approach is still key here, and Rackspace Technology uses this approach when other business priorities make full refactoring a challenge.

This is a common path to take as it requires less engineering effort than a full refactor and still allows for many of the benefits of the cloud. Obstacles to this process may come in the form of licensing limitations for the workload, the application behavior or installation, or the obvious factors of limited time and resources.

For example, it may be relatively simple to automate the installation and configuration of a web application running in Apache® on Ubuntu® simply because of the nature of the operating system, its scriptability and the configuration processes required. However, an off-the-shelf application running on Windows may be packaged in such a way that the installation cannot be fully automated despite all the neat tricks in your toolbelt, or may require that the vendor log into the server after installation to manually license the application (yes, this happens), or any number of factors that make it impossible to automate the process end-to-end. Workarounds can likely be identified for some of these issues, but it all depends on the amount of time and energy the organisation wants to sink into this process before cutting losses and choosing another path.



3. Replatform for partial automation

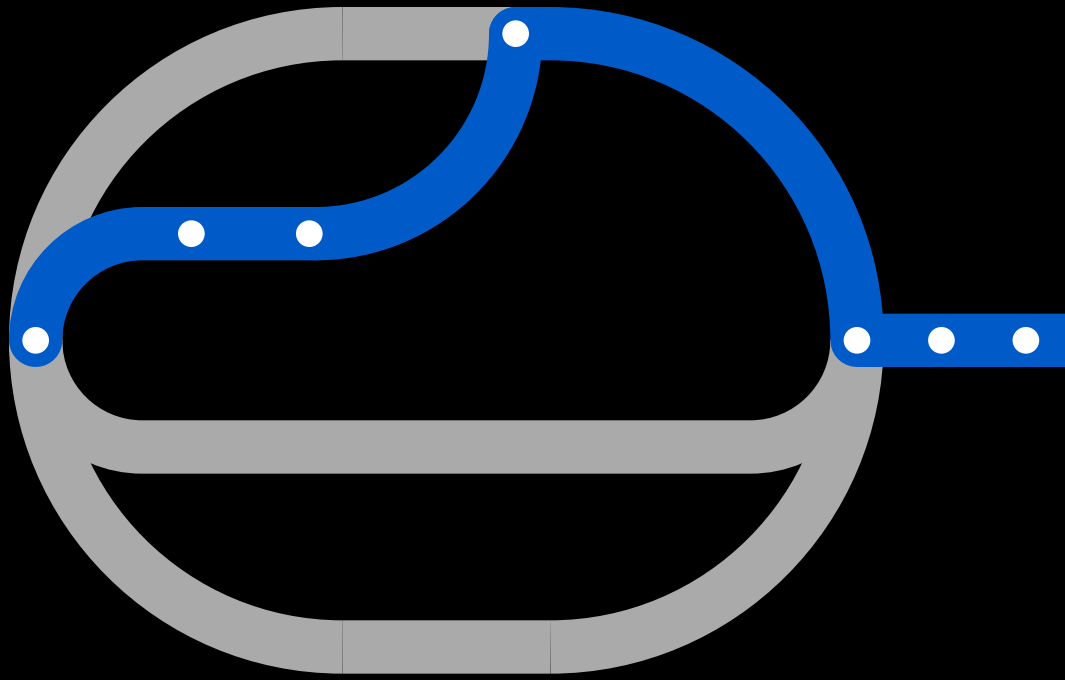
In situations where fully automated processes simply can't be achieved, as in the lift and shift pathway, the concepts of "infrastructure as code" and partial automation step in to bridge the gap.

Infrastructure code can be written to deploy resources such as servers, load balancers, etc. but in this path manual intervention is required to install and configure the workloads, databases and glue in between those components. An example of this might be a common workload "farm" at an enterprise. Terraform or a hyperscaler equivalent such as AWS CloudFormation may be written to deploy a collection of servers in the server farm, database servers running a given version of the database engine, load balancers pre-configured with a few listener rules and security groups to ensure connectivity between resources. From here, a systems administrator or application owner would log into the servers, run installation scripts if possible, manually configure connectivity between nodes by including static IP addresses or hostnames, for example.

This is still a beneficial path in the event that the migration path is very large. Often times this automation, while not end-to-end, can still be leveraged for its generality. The same infrastructure code to launch an eight-node Windows farm with a load balancer can likely be used to deploy a six-node Linux® cluster intended for a Cassandra installation, with only a few tweaks to the number of nodes, the machine image used, and a yes/no option for the load balancer. Even incorporating this level of automation can help accelerate additional effort in the future, and provide a starting point for later ventures into the more complex automated paths.

The downside of this path is that operations are performed manually. This inherently means that instead of treating your services like nameless, faceless resources that do their job on their own, you're stuck in the older data-centre-centric model of naming your servers, taking care of them and troubleshooting them when they fail. While this may sometimes be a necessary approach, Rackspace Technology still tries to avoid it when possible. Our belief is that using a model that acts as a data centre prevents you from getting the exciting futuristic benefits of the cloud, impacting the overall value. Instead, we've done more of just moving our datacentre to another location. But we have a lot of exciting scripts that we've written, and can hopefully use what we've learned writing them to move forward to more automated processes.

rackspace technology®



1. **Determine new platform**
2. **Modify underlying infrastructure**
3. **Use migration tools**
4. **Validation**
5. **Transition**
6. **Production**

Learn how the Central and North West London NHS Foundation Trust (CNWL) replatformed to improve application performance and enabled a fast shift to remote work.

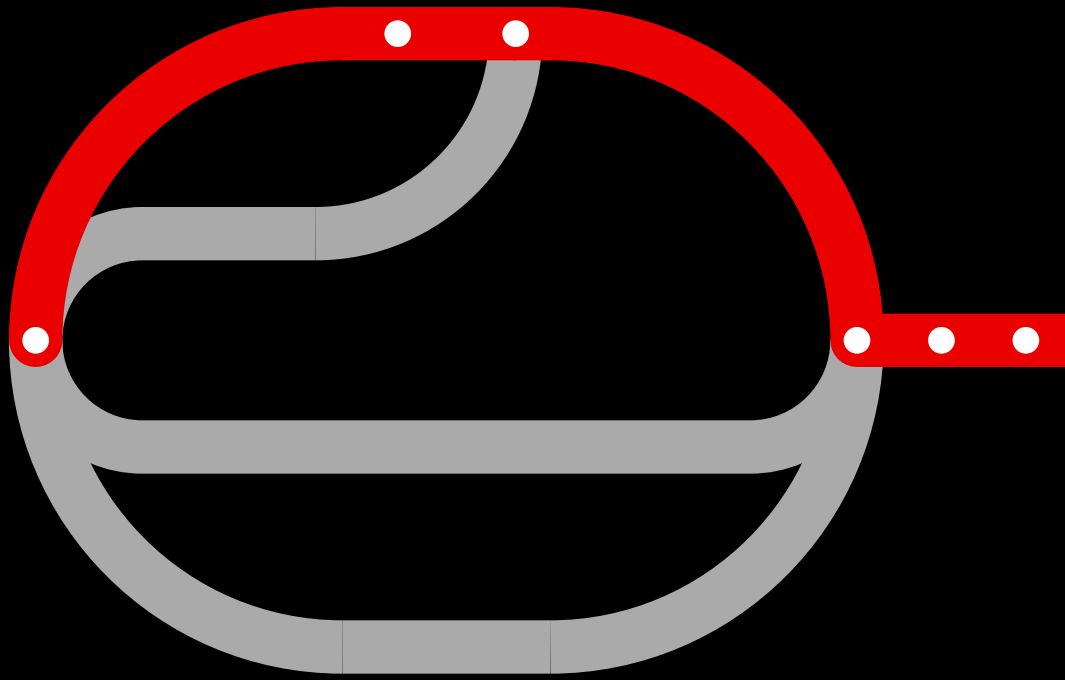
4. Rehost

The rehost migration path uses a third-party tool to create a duplicate machine currently running on-premises or in colocation, sends it to the cloud as a machine image, and then launches the server in the cloud. It is essentially an identical, block-for-block clone of the original machine. Often the tool used for this process provides simple interfaces for managing this process, and can typically synchronise data changes as they happen on the live machine up to the visualised image. This is a particularly ideal choice with the “black box” situation, which is a server that has been around since the dawn of the company running important business logic. As long as the baseline requirements are met for the migration tool, you can ship it and turn it on, and it’ll be the same box as on-premises. When it comes to speed and minimal level of expertise required, this is generally the easiest way to go, with some significant caveats.

The tools that offer this migration path are very limited. They can achieve their task (usually) but can’t handle variation on that process. They also cannot be automated, as the target demographic tends to prefer Graphical User Interfaces over scripting languages or command-line options, so commercially this is where the vendors spend their development energy. These tools can often be very expensive, and require licenses to be purchased per server slated for migration.

Data synchronisation may not be able to keep up with highly active changes, and we’ve seen multiple cases where a database was started with this process and could never keep up with shipping the delta up to the cloud, eventually falling farther and farther behind until a new migration strategy had to be implemented. Finally, sometimes things just don’t work. For example, the image doesn’t boot for mysterious reasons, and nobody (even the vendor) seems to know why, and you just have to try again or find another solution. At the end of the day, Rackspace Technology recommends avoiding a lift-and-shift migration at all costs because not only will there be inevitable problems in your migration, but you won’t be able to utilise the full potential of the cloud.

Most commonly, the issue with this migration path tends to be the inability to test effectively in Windows environments, tied to the use of Active Directory. When a Windows instance joins Active Directory, it defines several factors about itself that just aren’t very cloudy: hostnames often need to stay static for DNS and application-configuration purposes, the operating system sets several unique identifiers in the Windows registry that specifically define what this machine is and what it does, for example.



1. **Automate**
2. **Use migration tools**
3. **Validation**
4. **Transition**
5. **Production**

Learn how Dole Europe managed to lift and shift 50% of its workloads to Azure.

In each of these examples, if a second machine were to come up at the same time and connect to Active Directory, there would be duplicate machines and Active Directory wouldn't work correctly. In addition, SysAdmins would have difficulty troubleshooting the issues, and there might be production outages. This means that testing of the generated machine image should happen in a bubble without connectivity to Active Directory — which usually means your application isn't working, and your test isn't valid. All testing should happen during cutover, while the production instance is down and disconnected, which can lead to even longer outage windows.

Honourable mention: data migration

Downtime, and your inaccessibility to data, is the critical factor in each of these migration scenarios. With very large recovery time objectives (RTO), a database can be taken offline, backed up, the backup sent to a new database server in, and traffic shifted to this new server. With very low RTO, the ideal scenario is to attempt to expand the database cluster by adding a replica in the cloud, ensuring streaming replication in near real-time, and then cutting to the replica using the database's own failover mechanisms. These capabilities are limited to the database engine in use, and potentially the licensing level of that database, so always be sure to perform discovery ahead of time for any of these factors that may come into play when making a decision to migrate a database.

Tools that convert data from one database engine to another do exist, allowing for migration out of one engine and into a comparable platform (MS SQL to PostgreSQL is very common). However, these tools tend to manipulate the data, and may have technical limitations that make the data unusable in the new engine, so this must be tested ahead of time. Another issue with this method is that it requires testing of client applications to ensure proper connectivity to the new database engine.

Conclusion

The overall concept of these points is simply that there is no single tool for a migration, and care must be taken to identify the proper path for each workload to align it with business objectives. At Rackspace Technology, we always recommend automating as much as possible, because reusability is key when it comes to any technical process.

Customer First. Cloud First. An outcome focused strategy for the future

We take the complications out of cloud migration. After all, we believe that a well-integrated multicloud environment harmonizes and enhances the power of your data, applications and security solutions.

Discover cloud migration services from Rackspace Technology. Our experienced Professional Services team is staffed with cloud migration experts who are ready to help you plan and execute your migration.

Learn more at www.rackspace.com/services/consulting

rackspace technology®

About Rackspace Technology

Rackspace Technology is the multicloud solutions expert. We combine our expertise with the world's leading technologies — across applications, data and security — to deliver end-to-end solutions. We have a proven record of advising customers based on their business challenges, designing solutions that scale, building and managing those solutions, and optimising returns into the future.

As a global, multicloud technology services pioneer, we deliver innovative cloud capabilities to help customers build new revenue streams, increase efficiency and create incredible experiences. Recognised as a best place to work, year after year, by Fortune, Forbes, Great Places to Work and Glassdoor, we attract and develop world-class talent to deliver the best expertise to our customers. Everything we do is underpinned by an obsession with our customers' success — our Fanatical Experience® — so they can work faster, smarter and stay ahead of what's next.

Learn more at www.rackspace.com or call:

DACH: +49 (0)800 723 87 49
MEA: +9714 3999496
NE: +31 20 753 3200
UK: +44 (0)20 8734 8107

Rackspace-White-Paper-Four-Paths-to-Consider-for-a-Successful-Migration-PUB-TSK-7624 :: October 17, 2022