TechTarget | SearchCloudComputing

PRO+ Content

Cloud Computing: Market Insights and Best Practices for 2020

In this e-guide

In this e-guide:

Cloud computing is only becoming a larger and larger umbrella term that encapsulates platforms, software, infrastructures, containers, and many, many other technologies – so, keeping up with the market can be critical for your business.

To reflect the ever-growing market, this guide spans various cloud topics, including open source monitoring tools, refactoring strategies, tools for Kubernetes, assessments of various cloud platforms, and more. Read on to learn the best practices for cloud computing, and important trends you'll need to be aware of in 2020.

In this e-guide

7 key characteristics of cloud computing

Trevor Jones, Site Editor

Cloud entails an ever-expanding list of tools and techniques, but the key characteristics of cloud computing have always remained the same.

AWS was the first to popularize cloud computing as an alternative to on-premises infrastructure when it began selling compute and storage instances in 2006. Google and Microsoft followed soon after. Today, cloud computing extends from IaaS to SaaS and everything in between, including AI, containers, serverless, databases, IoT, dedicated networking, analytics, business apps and much more.

Each subset has its own benefits and challenges, but there are several core features that underpin all of them. Explore these 7 key characteristics of cloud computing that help explain why it's the go-to destination for building and deploying modern applications.

1. **On-demand computing**

AWS, Microsoft Azure, Google Cloud and other public cloud platforms make resources available to users at the click of a button or API call. These vendors have massive amounts of compute and storage assets at the ready, inside data centers all over the world. This represents a radical departure for IT teams accustomed to an on-premises procurement process that can take months to complete.

2. **Self-service provisioning**

In this e-guide

This characteristic of cloud computing goes hand in hand with on-demand computing. Instead of waiting for new servers to be delivered to a private data center, developers can select the resources and tools they need – typically through a cloud provider's self-service portal -- and build right away. An admin sets policies to limit what IT and development teams can run, but within those guardrails, they have the freedom to build, test and deploy apps as they see fit.

3. **Resource pooling**

Public cloud providers rely on multi-tenant architectures to accommodate more users at the same time. Customers' workloads are abstracted from the hardware and underlying software, which serve multiple users on the same host. Cloud providers increasingly rely on custom hardware and abstraction layers to improve security and speed users' access to resources.

4. **Scalability**

Resource pooling enables scalability for cloud providers and users because compute, storage and networking assets can be added or removed as needed. This helps enterprise IT teams optimize their cloud-hosted workloads and avoid end-user bottlenecks. Clouds can scale vertically or horizontally, and providers offer automation software to handle dynamic scaling for users.
Traditional, on-premises architectures can't scale as easily. Typically, enterprises have to plan for peak capacity and have those extra resources sit idle during lulls in activity, which can rack up costs.

5. **Pay-per-use pricing**

This cloud computing characteristic shifts IT spending from Capex to Opex, as providers offer per-second billing. Though this can generally be seen as a positive, IT teams need to be

In this e-guide

careful since their resource needs likely aren't static. VMs should be right-sized, turned off while not in use, or scaled down as conditions dictate. Otherwise, organizations waste money and can end up with sticker shock when they receive their monthly bill.

This pricing model was once the only way to pay for cloud, but vendors have since added various pricing plans that often provide cheaper costs in exchange for longer-term commitments.

6. **Resiliency**

Cloud providers use a number of techniques to guard against downtime, such as minimizing regional dependencies to avoid single points of failure. Users can also extend their workloads across availability zones, which have redundant networks connecting multiple data centers in relatively close proximity. Some higher-level services automatically distribute workloads across availability zones.

Of course, these systems aren't foolproof. Outages occur and enterprises must have contingency plans in place. For some, that means extending workloads across isolated regions or even different platforms, though that can come with a hefty price tag and increased complexity.

7. **Security**

To date, there have been no known breaches of the underlying resources of the major cloud platforms. And while many enterprises balked at migrating workloads because of security fears, those concerns have largely subsided, partly due to the benefits of the above characteristics of cloud computing. Cloud vendors employ some of the best security experts in the world and are generally better equipped to handle threats than most corporate IT teams. In fact, some of the biggest financial firms in the world say the cloud is a security asset.

However, this doesn't absolve users of their duties. Public cloud providers follow the shared-responsibility model -- they tend to the security of the platform and the users handle their own apps that sit on top. Failure to fully grasp those delineations has led to some embarrassing, high-profile exposures of sensitive corporate data.

▼ Next Article

In this e-guide

What you need to know about Google Cloud Anthos

Kurt Marko, Consultant

Not all enterprises want to abandon their existing infrastructure or are comfortable running all of their applications on shared services. To address these concerns, top public cloud providers have hybrid offerings that extend their services on premises.

A key challenge for providers is adapting cloud services to host legacy systems that were designed when each application had its own server. Typically, this has been addressed through virtualization, but the strategy of treating the cloud as a big VM server farm sacrifices benefits of a cloud infrastructure, like efficiency, scalability and adaptability.

Google Cloud Anthos does not force organizations to make an all-or-nothing decision on shared services. Instead, it gives customers the option to use cloud technology -- namely, containers and Kubernetes clusters -- on existing internal hardware. With a consistent design and set of services for both on-premises and in-cloud deployments, Anthos gives organizations the freedom to choose where to deploy particular applications and migrate workloads between environments.

Google Cloud Anthos enables enterprises to use container clusters, instead of cloud VMs, to bridge the gap between legacy software and cloud hardware.

Anthos 101

Google Cloud Anthos, previously known as Google Cloud Services Platform, is a cloud-agnostic container environment. It uses Kubernetes for workload orchestration and Istio for app and microservices traffic management, as well as routing.

Anthos is purely a software product that can run on existing hardware. Google has partnered with hardware providers, such as Cisco, Dell EMC, Hewlett Packard Enterprise, NetApp and Robin.io, to deliver prepackaged Anthos systems, including on hyper-converged appliances. However, it can run on any servers capable of hosting Kubernetes clusters.

Anthos aims to solve the problem of containerizing legacy applications by including migration software that transforms VM images into containers before deploying them onto Anthos. Lastly, the service is rounded out by a host of monitoring, security, authorization and application management tools.

Core components, features and pricing

The foundation of Anthos is a container cluster managed by Google Kubernetes Engine (GKE). To accommodate hybrid environments, Anthos supports both the GKE managed container service and a GKE On-Prem environment that bundles the same set of management and security features.

Atop the Kubernetes base, Anthos adds the following core components:

In this e-guide

Primary components of Google Anthos

COMPONENT	CLOUD	ON PREMISES
Google Kubernetes Engine	GKE	GKE On-Prem (1.0)
Multicluster management	Yes	Yes
Configuration management	Anthos Config Management (1.0)	Anthos Config Management (1.0)
Migration	Migrate for Anthos	N/A
Service mesh	Istio on GKE, Traffic Director	Istio OSS (1.1.7)
Logging and monitoring	Stackdriver Logging, Stackdriver Monitoring, Alerting with Stackdriver	Stackdriver for system components
Marketplace	Kubernetes applications in GCP Marketplace	Kubernetes applications in GCP Marketplace

Take a closer look at the seven main components of Google Cloud Anthos.

1. **Anthos Config Management:** This component provides the tools required to set up and administer multiple Kubernetes clusters, while maintaining a consistent set of network and security policies across environments. It manages clusters via configurations that are stored in repositories such as GitHub or Google Cloud Source Repositories.
2. **Istio on GKE:** This service mesh securely connects clients to containerized services and applications while managing traffic flows between microservices. It also enforces security and usage policies.
3. **Traffic Director:** This traffic control plane for service meshes adds multi-region load balancing, health checking, demand-based autoscaling and more. It uses standard APIs that enable it to work with sidecar proxies, like Envoy.

4. **Stackdriver:** Google Cloud Platform's (GCP) monitoring, management and debugging service provides a central repository for network, application and infrastructure telemetry. Google offers two support options: Legacy Stackdriver and Stackdriver Kubernetes Engine Monitoring.
5. **Migrate for Anthos:** This tool automates the process of migrating legacy virtualized workloads to GKE containers. It can also convert workloads from VMware vSphere, Amazon EC2 and Microsoft Azure VMs.
6. **Cloud Run:** This serverless platform can run event-driven workloads and deploy containers to a GKE cluster. It enables apps to invoke container-run functions without configuring servers, and it also automatically sizes compute resources according to workload demands.
7. **GCP Marketplace:** Anthos users can access prebuilt Kubernetes applications and development stacks on GCP Marketplace and automate container development processes using CodeBuild.

Additionally, Apigee -- GCP's API management service -- simplifies the process of exposing programming interfaces for services running on GKE.

Usage

Google uses containers to deliver a consistent hybrid cloud environment, and enterprises can use Google Cloud Anthos' various components to manage and coordinate their on-prem and cloud workloads. However, we still don't know how the service works in practice since it's so new and there aren't independent case studies from enterprises using it in production.

Google has a lot experience with containers, including initially developing Kubernetes. If SAP HANA can run on a Kubernetes cluster, an enterprise app that can be virtualized should work with Anthos.

Organizations aggressively pursuing cloud-native applications with containers and microservices will find a lot to like with Anthos since it is built atop a strong container-as-a-service platform and provides a smooth migration path for legacy applications.

Competition

Google's hybrid strategy is decidedly different than that of AWS or Microsoft, which continue to use virtualization to maintain consistency between environments rather than containers.

For example, AWS partnered with VMware to provide an enterprise virtualization platform on AWS infrastructure. Organizations that have standardized on VMware and want to preserve that investment as long as possible will find VMware Cloud on AWS the best fit. AWS also has Outposts, which packages infrastructure and services inside private data centers and operates like Amazon's public cloud.

Similar to Outposts, Microsoft has developed a subset of Azure capable of running on premises via Azure Stack. Microsoft shops will likely find Azure and Azure Stack the most natural path to hybrid.

Next Article

In this e-guide

Key considerations for refactoring applications for the cloud

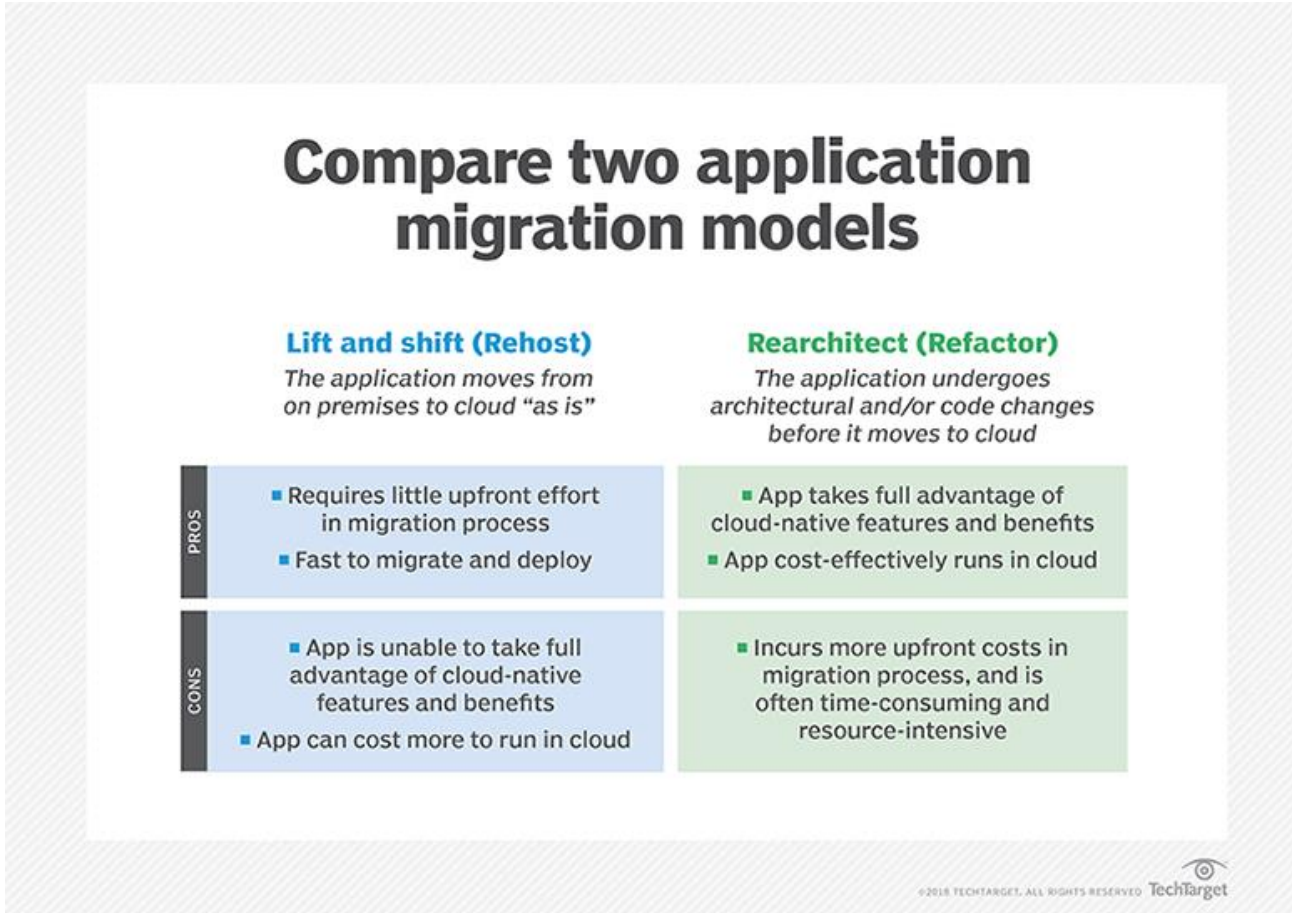
David Linthicum, Chief Cloud Strategy Officer

Organizations lift and shift their workloads to the cloud because it's cheaper and quicker than other migration techniques, but that doesn't mean it's always the best fit.

IT teams often face budget and time constraints, so they believe they have little choice. In addition, they may feel rushed to get applications up and running by cloud providers eager to bill these new workloads. But there are significant disadvantages to a lift-and-shift approach when compared to application refactoring, also known as *rearchitecting*.

It may be cheaper upfront to simply rehost your application and its data as is on the public cloud, but this approach could ultimately cost more than it would to run a cloud-native app instead. There may also be performance issues caused by changes in the software architecture, missed software bugs and an inability to properly utilize cloud vendors' native services for monitoring, security and governance.

It's often best to refactor an application as part of a migration, but sometimes, organizations do so retroactively. This can happen when performance fails to meet expectations after a lift and shift, and tuning doesn't solve the problem. A migrated application may also benefit from refactoring when bills are unexpectedly high due to application or database inefficiencies or when security vulnerabilities arise because the application can't integrate with native security systems, such as identity and access management tools.



When and how do you retro-refactor?

Organizations must consider several factors before they decide to refactor their apps -- the most important of which is cost. If you can't make your money back from refactoring applications, then it should not be attempted. Consider the previously mentioned signs around cost, performance and security when you analyze the return on your investment.

Also, it's not like refactoring in-flight, during the cloud migration. You have an application that's in production currently, and you have to disrupt that production on the cloud, somewhat, when it's refactored. As a result, you disrupt the user twice.

There's also a great variety of refactoring tools to choose from, and an application's needs will vary depending on what programming languages and databases that app relies on. Generally speaking, however, categories of tools include anything that assists in designing and developing microservices that utilize cloud-native APIs. Moreover, container development and Kubernetes deployment and operations are also a common way to refactor.

Organizations also have several ways to refactor their applications for the cloud.

A **complete refactor** is when more than 50% of the code is changed and the database is updated to utilize as many cloud-native features as required by the application. This strategy can improve performance, operations costs and IT teams' ability to meet the needs of the business. However, the process could be too costly or complex, and it can introduce bugs.

Minimum viable refactoring prioritizes speed and efficiency, as it requires only slight changes to the application. Users who take this approach often incorporate cloud-native security, management and perhaps a public cloud database into their migrated workload.

In this e-guide

This approach doesn't offer as many cloud-native benefits as a complete refactor, but it may have some legs. For example, Philip Potloff, head of enterprise strategy at AWS, has a two-week rule for refactoring applications. This technique, he suggested, gives cloud migration teams enough time to refactor what is determined to be the "most problematic compatibility issues with the cloud."

Generally speaking, most applications and data sets that can move to a public cloud require at least some refactoring.

Containerization refactoring is done when applications are moved into containers with minimal modifications. The applications exist within the containers, which enables users to incorporate cloud-native features and improve portability. Enterprises have found this option is more work than expected because of the learning curve that comes with adapting to these new tools. With that said, costs and refactoring times continue to go down due to the popularity of containers and their growing ecosystems.

A **serverless application refactor** has similar issues as containerization in that it changes the development and operations platform, which requires learning new tools and skills. Some modifications are required to make the application work effectively and take advantage of serverless systems on the public cloud. For example, serverless platforms support most languages, but they don't support everything; the same goes for databases.

Unlike containers, serverless platforms don't provide portability, so lock-in is a major downside to this approach.

In this e-guide

Where to go from here?

Enterprises should bear in mind that, generally speaking, most applications and data sets that can move to a public cloud require at least some refactoring. However, this should not involve major surgery that lasts for months, because most of those efforts won't be economically viable.

Users should at least do minimum viable refactoring for most of the cloud-hosted application. Potloff's two-week target may be too aggressive for many IT teams, but there should at least be some hard-and-fast deadlines in place. A targeted timeline will keep teams out of trouble and ensure that most of the refactoring work will have a quicker return on their investment.

Next Article

In this e-guide

5 open source cloud monitoring tools to evaluate in 2019

Chris Tozzi, Analyst

The open source software ecosystem has grown bigger and better over the last decade, and cloud monitoring tools are no exception. In 2019, enterprises will have a wide range of options for open source tools to monitor cloud apps.

That's good news for developers and admins, since the native monitoring tools available on public cloud platforms, such as AWS and Azure, aren't always enough to properly monitor and manage cloud apps -- particularly those in multi-cloud or private cloud environments.

Ranging from some lesser-known options to those many developers already know and use, here's a look at five open source cloud monitoring tools to consider in 2019.

Riemann

Riemann provides a single, straightforward tool to monitor distributed applications and infrastructure. The open source software enables developers to define various types of events to monitor, as well as *streams* that generate alerts when a particular type of event occurs. Developers can configure streams to send email notifications or alerts via Slack about events.

Riemann builds an index of all of the services it monitors and provides a dashboard to visualize that data. Developers can also set up Riemann to feed data to a third-party open source data visualization tool, such as Grafana.

For some developers, Riemann's native feature set might seem somewhat limited. For example, there's not a lot of room for customization within the dashboard. However, Riemann can integrate with a range of external tools that are more customizable.

cAdvisor

Another open source monitoring tool that's lesser-known, but worth a look, is cAdvisor. Short for Container Advisor, it was one of the open source monitoring tools purpose-built for containerized applications. While it won't monitor other types cloud apps, cAdvisor runs as a container itself, so users just need to spin up a new container to deploy it.

CAdvisor provides a browser-based graphical interface for data visualization, and developers can configure the tool via a command line. Compared to some commercial cloud monitoring tools, cAdvisor features might seem relatively basic; however, as one the only open source monitoring tools built specifically for containers, it's worth a look for teams that use Docker to host apps in the cloud.

Elasticsearch

Some enterprises will already be familiar with Elasticsearch, an open source tool to search through large volumes data.

While Elasticsearch isn't always put in the family open source cloud monitoring tools, its search capabilities can come in handy for that use. Teams will need to integrate the tool with others, such as Kibana and Logstash, to collect and visualize the monitoring data that Elasticsearch will process. Still, as part a larger stack, Elasticsearch can provide powerful search functionality to help admins and developers better understand their cloud monitoring data. It's also designed for massive scalability, so teams can use it to monitor a handful, or thousands, application instances without having to change tool sets.

When used alongside other monitoring tools, Elasticsearch is essentially an open source alternative to Splunk. That's not to say Elasticsearch can be a full replacement for Splunk, but in some cases, it is a viable and less costly alternative.

Graphite

Graphite is a popular open source tool -- set to remain so in 2019 -- to help monitor cloud apps and infrastructure.

The scope Graphite is fairly narrow; it enables users to take time series data and visualize it. It doesn't collect data or store it persistently, but enterprises can integrate Graphite with a variety other tools, including Riemann, to perform some these tasks. Because this, Graphite isn't an all-in-one cloud app monitoring tool. Still, it serves as a straightforward and highly extensible visualization tool for monitoring data.

Prometheus

For many admins and developers, Prometheus is one the more familiar open source cloud monitoring tools.

Prometheus is feature-rich and offers a range of customization options. It integrates with third-party visualization tools, including Grafana, and also provides a native visualization engine. It offers an API to send data to external tools, and users can prepare monitoring data for interpretation using a special query language called PromQL.

While Prometheus is one of the most robust open source cloud monitoring tools available today, there are certain things it does not do. For example, it's not a log management tool, it doesn't provide automated anomaly detection and its native visualizations are relatively basic -- though, again, integration with other tools can extend users' visualization options.

Next Article

In this e-guide

Compare tools for multi-cloud Kubernetes management

Tom Nolle, President

Kubernetes continues to evolve in many directions, which can make it difficult to understand its relationship with other IT trends -- especially multi-cloud.

Users can create a Kubernetes cluster and deploy containers inside VMs on any public cloud or on premises, but they still need to manage scaling and resiliency within those environments. Cloud providers offer managed services to address these problems, but those tools aren't built for a multi-cloud Kubernetes architecture.

Cloud providers' managed Kubernetes services handle resource deployment and management, including load balancing and network connections. But organizations need a Kubernetes framework that can cross those lifecycle management "islands" if applications run across cloud boundaries.

Popular tools for multi-cloud Kubernetes deployments

Users who want to deploy a multi-cloud Kubernetes strategy do have options. Here are some of the tools available to make container architectures work across multiple platforms:

NetApp Kubernetes Service -- formerly StackPointCloud -- is a well-known and mature service that provides a common administration console for multiple Kubernetes deployments. Each cloud represents a separate cluster, and you can spin up a cluster in any of the popular public clouds. However, it doesn't support on-premises Kubernetes. If a multi-cloud deployment includes a private data center, the cloud and data center workloads need to remain separate.

Cloudify offers a higher-level, model-driven orchestration tool that's open source and can deploy Kubernetes on multiple clouds, as well as on bare metal or inside VMs. Cloudify doesn't augment Kubernetes; instead, it treats Kubernetes and any cloud provider's Kubernetes service as a class of resource. This makes Cloudify a good tool to harmonize different hosting approaches, and not just different cloud providers. Still, some users won't like the additional layer of abstraction.

Terraform is an infrastructure as code (IaC) tool that provides a common hosting framework. It creates a single virtual pool of hosting resources for Kubernetes on any number of public clouds or private data centers. However, the advanced features in cloud providers' managed Kubernetes services don't translate to Terraform, which can limit what you can do on a given platform.

Rancher is a Kubernetes-centric framework that works with bare metal, VMs, on premises and on multiple public clouds. Rancher creates a three-level architecture, with application workload management at the top and unified cluster management in the middle. The bottom layer consists of the various public cloud Kubernetes engines and Rancher's own Kubernetes engine for private infrastructure. Users can opt for cloud providers' managed Kubernetes offerings or handle management on their own.

Platform9 Managed Kubernetes is a SaaS approach to hybrid and multi-cloud Kubernetes. It works on all the popular cloud platforms, as well as on an enterprise's own servers. It's a "bring-your-own-infrastructure" approach that handles all hosting resources through a central management console and adapts to virtually any hosting environment. Platform9 deploys

VMs, containers and even serverless, so it's very flexible. Many observers see Platform9 as a competitor to Rancher.

Red Hat OpenShift with Tectonic combines a Kubernetes-centric vision of abstract infrastructure with resource pools that can span public cloud and on-premises hosting. OpenShift takes an on-premises-centric view of multi-cloud and hybrid cloud, rather than a SaaS model, because it's based on tools an IT pro would be accustomed to inside their own data center. The OpenShift-Tectonic combination is new, so check the progress of the integration and the feature enhancement plans regularly when you consider your options.

Juke, from HTBase, which is now owned by Juniper Networks, extends both the control and data planes of a Kubernetes deployment, so it provides the network layer needed to create a uniform deployment framework across clouds and data centers. Juke has strong support for multi-tenancy, which makes it an attractive option for cloud providers and a good choice for users who need significant application isolation for governance or security.

Cloud's influence on Kubernetes

Cloud-based managed Kubernetes continues to shape the trajectory of Kubernetes as a whole. This can be seen through the various add-ons that can accommodate multi- and hybrid-cloud deployments, including Kubernetes plugins from vendors such as Red Hat, and the emergence of Isto as a service mesh.

Moreover, organizations are now forced to think of Kubernetes orchestration as the management of various installations, as opposed to the management of different Kubernetes clusters. In that context, the various clouds become those installations and act as extensions of your resources. A hierarchical model of Kubernetes is emerging, with lower and higher layers that work together and span all kinds of infrastructure.

In this e-guide

Kubernetes' strength has always been its open framework and APIs that enable so much integration. That integration process continues to redefine Kubernetes and its ecosystem, which means multi-cloud Kubernetes support, and the managed Kubernetes services from cloud providers, will continue to evolve. Users will have to carefully track the progress of their chosen tools, even after they've made their choice.