CONTAINERS AS A SERVICE (CAAS) EXPLAINED



Containerization is a hot topic that has drawn much attention among developers looking to build portable application components for multi-cloud infrastructure environments. So, it's no surprise that a subscription-based cloud model has emerged: containers as a service (CaaS).

What is containers as a service?

Containers as a service (CaaS) is a subscription-based cloud service model that allows you to manage containers, applications, and clusters using APIs, container-based virtualization, or Web portals.

The service helps to streamline the process of developing and managing containers within software-defined infrastructure deployed on-premise or in cloud environments.

The CaaS provider offers the orchestration platform that allows users to develop containerized applications. The cloud-native application components can be deployed and run from a cloud environment while users subscribe for CaaS resources such as scheduling capabilities, load balancing, and compute instances.

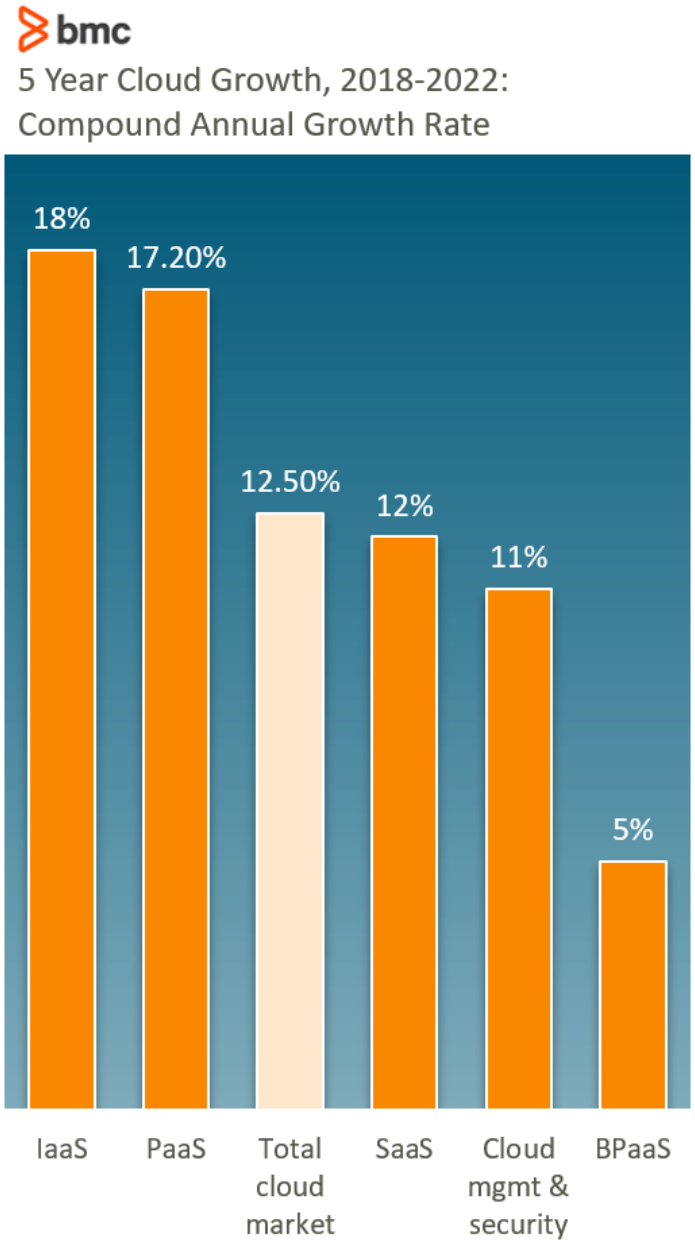
The most popular orchestration technologies used within the CaaS framework are:

- Docker
- Kubernetes
- Data Center Operating System (DC/OS)

CaaS vs IaaS

Container as a Service is a subset of the Infrastructure as a Service (IaaS) cloud delivery model that is growing in popularity. The ability to consume scalable data center infrastructure resources as an operating cost (OpEx) on a subscription basis is driving rapid investments in cloud technologies:

- The global cloud computing market is expected to reach over $1 trillion by the year 2026.
- 80% of all organizations are expected to adopt cloud infrastructure solutions by 2025.



How CaaS works

A container is a logical packaging of the IT infrastructure environment that is required to run an application. Similar to virtual machines (VMs), the packaging includes all the components that allow the container package to operate applications in isolation, including:

- Libraries
- Configuration files
- Dependencies

Containers are different from VMs in that containerization virtualizes at the Operating System (OS) level.

The primary resource of a CaaS service is the container itself, whereas an IaaS service usually offers VMs or bare metal servers. Containers are important for quickly building applications with microservices since application components are portable across cloud environments. As a result, a containerized application development process:

- Supports a multi-cloud strategy
- Optimizes operating costs
- Reduces overhead costs, such as licensing and operating fees

CaaS automates the process of hosting and deploying container technologies in highly available cloud environments. CaaS differs from the Platform as a Service (PaaS) paradigm, since CaaS is not associated with a specific code stack ecosystem or has dependencies on language runtime and databases at the application level.

In essence, CaaS doesn't suffer from the "it runs on my machine" problem, unlike PaaS services.

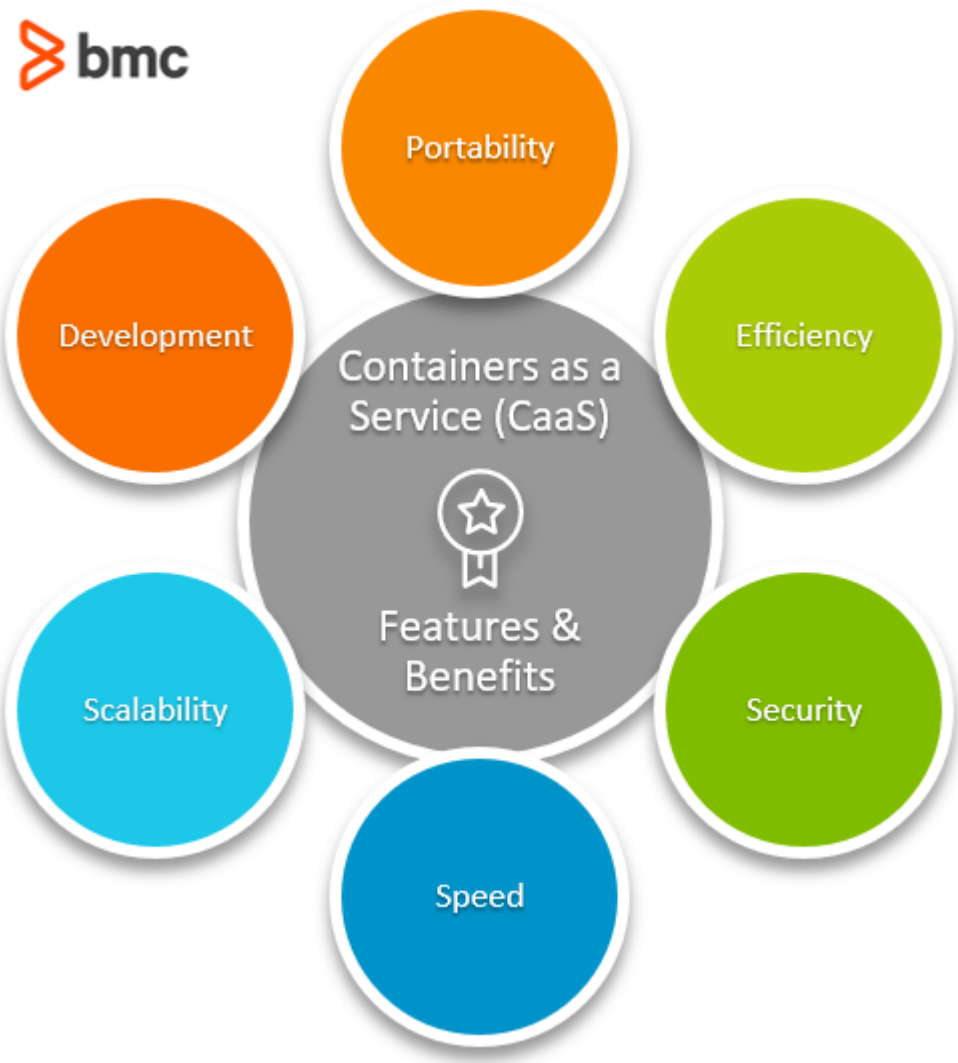
Features & benefits of CaaS for DevOps

Enterprises and DevOps organizations take advantage of CaaS solutions to:

- Accelerate the software development process
- Deploy innovative solutions in the cloud at scale

Software Development Lifecycle (SDLC) teams are able to release software products faster while reducing the resources, bottlenecks, and inefficient waste processes typical of software development and deployment. Surely, containers support and contribute to true business transformation.

These advantages are achieved thanks to these features and benefits of CaaS solutions:



Portability

Containers include all the app dependencies, libraries, and configuration files necessary to run the app in an isolated environment, decoupled from the underlying infrastructure. Users can switch to different cloud environments dynamically and yet reliably launch the applications as cloud-native technologies.

Efficiency

Containers require fewer files to run the applications and they run on a shared OS. The starting time for a container is a few minutes and the total size volume is within the megabyte range, unlike VMs, which require files in the gigabyte size range.

Security

Containers are virtually isolated from the underlying infrastructure and from other containers that may hold the components of the same application.

If one container component is compromised, the damage is contained, and other application components remain secure.

Speed

Provisioning and managing CaaS resources can be done with a few simple clicks—or you can automate it altogether.

The process takes only a few seconds, which means that DevOps organizations operating a high-speed SDLC process can develop new builds, fix bugs, or add features by provisioning new containers on a whim.

Scalability

The service allows users to scale container resources horizontally.

Multiple identical clusters can help manage peak workloads only when necessary to optimize resources utilization and cost investment. Automated provisioning and scheduling services help turn off the instances when not in use.

Development

The scalability, automated provisioning, and resource management capabilities of Container as a Service offerings help streamline the development process. Inconsistencies in the infrastructure environment are eliminated as developers can modify and provision containers at scale.

Popularity of containers

IaaS estimated as one of the fastest growing service delivery models, with its market is expected to reach $82 billion within the next two years. With this growth and the increasingly popularity and ease of containers, we have a hunch that containers as a service will really take off.

Related reading

- BMC DevOps Blog
- Docker Security: 14 Best Practices for Securing Docker Containers
- 15 Best Practices for Building a Microservices Architecture
- Containerized Machine Learning: An Intro to ML in Containers
- State of Containers: A Report Summary