CHAPTER 7

Cloud Computing

We discussed in the earlier chapters that in a distributed system, there are two parts: client and server. Traditionally, corporations have managed their back-end servers on their own at their physical premise. However, there is a trend to consolidate these resources and services elsewhere (the cloud) on a network. These services can be used by the client systems as needed, and the resources can be remotely shared and optimized. The services are provided and managed by "cloud service providers" (CSPs). In this chapter, we'll discuss different cloud computing models, their implications, and trade-offs. We'll follow that up with different deployment configurations and consideration for developing and deploying portable and interoperable cloud solutions.

Note Simply speaking, cloud computing is a mechanism that delivers computing services over the Internet ("the cloud") to offer faster innovation and dynamic scaling. It can help lower the operating costs for many of the usage scenarios by means of more optimized resource utilization.

Figure 7-1 illustrates cloud computing. Essentially, the infrastructure, platform, and services are hosted in the cloud; and then customers can access these services over the Internet via various interfaces.

© Paul D. Crutcher, Neeraj Kumar Singh, and Peter Tiegs 2021
P. D. Crutcher et al., *Essential Computer Science*,
https://doi.org/10.1007/978-1-4842-7107-0_7

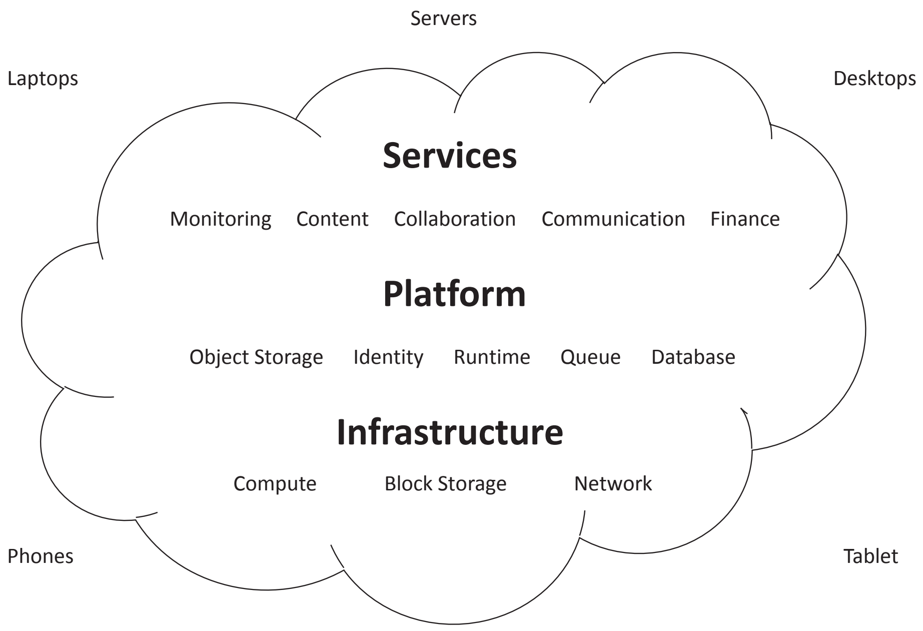

Figure 7-1. *Illustration of Cloud Computing*

Cloud Computing Models

There are different models of providing cloud computing services. Broadly speaking, these cloud computing service offerings fall into four categories: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), serverless (aka Function as a Service, or FaaS for short), and Software as a Service (SaaS). Serverless (aka FaaS), however, is usually considered the same as or an extension to PaaS and not treated as a separate model in some literature. These models are also referred to as a cloud computing pyramid or stack because they build on top of one another.

Figure 7-2 depicts how the various models stack on each other. In the following sections, we'll briefly discuss these models one by one.

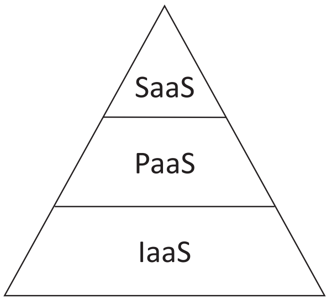

Figure 7-2. *Cloud Computing Pyramid or Stack*

IaaS

Infrastructure as a Service (IaaS) is the most basic and flexible model of cloud computing. Essentially, IaaS provides a virtualized computing infrastructure that is provisioned, managed, and accessed over the Internet. Virtualization is a mechanism of creating virtual computer hardware like CPU, storage, network, and so on. An IaaS provider manages the physical end of the infrastructure (compute, storage, memory, and network) in a shared data center and lets their customers customize and use those virtualized resources to suit their needs.

The other way to look at IaaS is that the cloud service customer (CSC) rents the hardware resources from a cloud service provider (CSP). The hardware is hosted and managed by the CSP. It is essentially like a customer getting a machine with the requested hardware – compute, storage, memory, and network – from the CSP accessible over the network. The customer is responsible for the rest of the infrastructure management (OS, software, security, etc.).

IaaS provides dynamic scaling up and down based on the demand that lets customers pay only for what they use. Because compute resources are subscribed, purchased, and used as a service, it helps customers avoid the upfront cost and complexity of procuring and managing their own physical servers and other related infrastructure. Each part of computing resources is offered as a separate service component, and then the customer can choose and assemble required compute resources. For instance, customers can choose the number of CPUs, memory, storage, and networks separately based on their need, and the CSP will assemble and make a virtual system with those chosen resources. The resources need to be subscribed and rented only for as long as needed. The cloud service provider manages the virtual hardware infrastructure, while customers purchase, install, configure, and manage their own software pieces like operating systems, middleware, and applications. Microsoft Azure, Amazon Web Services (AWS), and Google Compute Engine (GCE) are examples of IaaS.

IaaS services make it super quick and easy to get access to hardware resources; you don't have to procure, provision, and secure the hardware. Given the control and flexibility of software deployment in this environment, IaaS is the most efficient (in terms of time, cost, and flexibility) for exploration work. Also, owing to the flexibility, IaaS is always available for scenarios where the other services like PaaS and SaaS are unavailable.

It is also to be noted that when customers use more value-added cloud services from a CSP, they are charged extra for those value additions. For instance, PaaS is more value-added service than IaaS because CSPs are responsible for more services as part of PaaS than IaaS. So a corollary of that is IaaS will be cheaper than PaaS for same level of usage.

PaaS

Platform as a Service sits a little bit higher up the pyramid than IaaS. That means, as part of PaaS offering, the CSPs are responsible for, and maintain, more services than in an IaaS model. As part of PaaS, CSPs supply an on-demand environment for developing, testing, delivering, and managing software services. PaaS makes it easier for developers to quickly create solutions without worrying about setting up or managing the underlying infrastructure of hardware and software needed for development and deployment. AWS Elastic Beanstalk, Apache Stratos, and Google App Engine are some examples of PaaS.

PaaS offers everything that IaaS provides – that is, the underlying infrastructure as the service. However, in addition to the hardware, PaaS consists of middleware, framework, and other development and deployment tools. As part of PaaS, the cloud service providers provide these platform ingredients (tools and software). So the customers can focus on their development rather than trying to manage the SW and HW infrastructure.

Serverless

Serverless computing is an extension to PaaS. This is also known as “Function as a Service” (FaaS). FaaS allows customers to execute code when needed without having to allocate computing resources in advance. In other models like IaaS and PaaS, the user has to allocate the computing resources in advance. As with PaaS, the cloud provider manages the complete infrastructure. This allows the customer to focus on deploying application code in the form of “functions.” FaaS enables your functions to scale up or down automatically to meet demand, which makes FaaS an excellent fit for workloads that fluctuate in terms of resource requirement.

Essentially, serverless architectures are highly scalable and event driven, only using resources needed to fill the given demand. Customers only pay for the resources they consume; therefore, serverless (FaaS) is the truest form of “pay-as-you-use” cloud computing model. Some examples of serverless are AWS Lambda and Azure Functions.

The serverless usage model is best suited for burst, trigger-based usage scenarios and can support extremely high throughput. The customer does not have to care or preplan for the infrastructure. The CSP infrastructure will automagically provision the platform and deploy and run the code when there is a trigger. There are many organizations, for example, Thomson Reuters, the Coca-Cola Company, Netflix, and so on, that are already leveraging serverless effectively.

SaaS

Software as a service (SaaS) is the model where software applications are delivered for use as needed over the Internet. Applications are typically made available on a subscription basis. With SaaS, cloud providers host and manage the software application including the required infrastructure and maintenance, including software upgrades and security patching. Customers just use the software application, over the Internet, without worrying about any aspect of development, deployment, and maintenance of the software.

SaaS sits at the top of the pyramid, and for the majority, it is the most familiar form of cloud computing. Some examples of SaaS include Microsoft Office 365, Salesforce, and Gmail.

SaaS is where the customers are not bothered by or responsible for any other aspects of software except that it should be reliably available when needed in a secure fashion. The service provider is responsible for everything. This is the only practical model for individual end users. However, there are organizations that don't want to develop, deploy, and maintain their own software application for a specific purpose and so buy a subscription and let their employees use it. This allows organizations to focus on their core business rather than be distracted by other needs.

Comparison of Cloud Computing Models

Figure 7-3 shows you the split responsibilities between the cloud service provider and the cloud service customer across IaaS, PaaS, and SaaS. As it is evident, the management responsibility of the CSC goes up from IaaS to PaaS to SaaS. Roughly speaking, in the IaaS model, the user gets the hardware equivalent of compute resource and everything else is managed by the user, while in the SaaS case, pretty much everything is managed by the service provider and the user just needs to use the software.

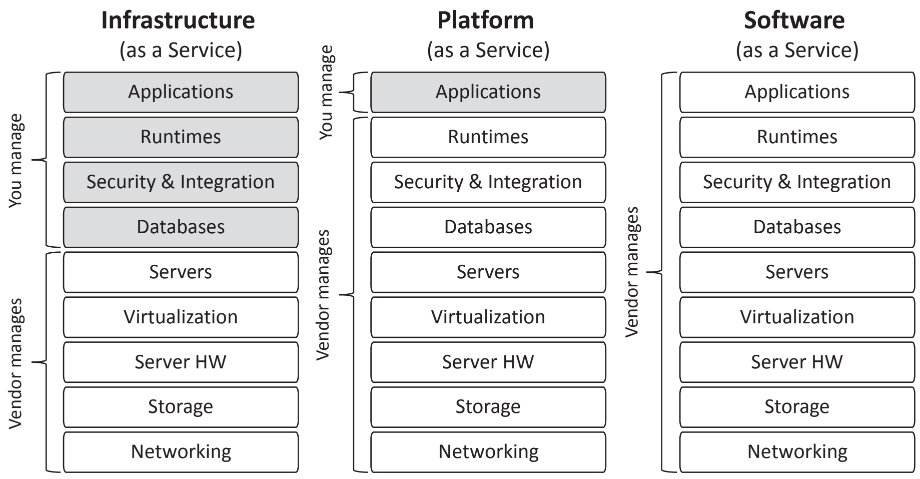

Figure 7-3. *Service Management Responsibility Chart Across IaaS, PaaS, and SaaS*

Benefits of Cloud Computing

As we've discussed, there is a lot of traction and movement to cloud computing. Organizations either have adopted or are in the process of defining the cloud strategy for optimal benefits. Much of traction is for good reason. Cloud computing offers organizations many benefits, which we will discuss next.

Cost

In the cloud computing world, the customers use computing resources provided by the cloud service provider. Cloud computing eliminates the capital expense of buying hardware and software, physical hosting of hardware, and setting up and running on-site data centers. The capital cost is completely taken away and born by the cloud service providers. The same is true for operating and management costs of the computing resources and other accessories like the electricity for power and cooling.

The customer must be aware that nothing comes for free. CSPs charge customers for the services they offer. Practically, in most of cases, the amortized cost for the customer will turn out cheaper using the cloud than hosting their own infrastructure. There are some scenarios where it could be costlier to use cloud services as compared to hosting one's own data center. However, even in those cases, there are other benefits of using cloud services vs. hosting one's own data center, which we discuss in the following sections. Finally, as this ecosystem evolves, we see that it is more and more likely that large corporations will have a split of cloud and on-premises compute infrastructure that can be combined, in a so-called hybrid cloud, to get the best of both worlds.

Scalability

Another key benefit of cloud computing service is the ability to scale dynamically. There are multiple use cases where the computing needs may be bursty in nature. For instance, during festival time, there could be a lot more load on the ecommerce websites than otherwise. Similarly, for a geography-specific service, there could be more load in daytime than at midnight. So dynamic scalability refers to the ability of cloud to deploy the right amount of resources like computing power, storage, and bandwidth as per demand.

The other very important side effect of dynamic scalability is that the customer need not plan for and buy worst-case workload. It just scales as needed. The CSPs do charge for enabling dynamic scaling; however, given the benefits, it is totally worth it.

Velocity

As the cloud computing resources are already pooled at the CSPs and services are provided on demand, practically all computing resources can be provisioned quickly, in minutes in fact. This facility is in complete contrast to the procuring hardware resource and deploying in the traditional data center world, which can take months, if not quarters, to complete. This capability enables great flexibility for customers and takes off the pressure of capacity planning.

Reliability and Availability

Cloud service providers provide robust solutions for data backup, disaster recovery, and business continuity in an easier and less expensive manner. The cloud service providers add redundancy and apply modern management tools to make the cloud computing resources and overall environment reliable and available.

Productivity

Cloud computing enables high productivity for customers. For instance, with a traditional data center, setting up and managing computing resource requires a lot of time-consuming chores: hardware setup, software patching, security updates, and so on. In the cloud computing world, these chores are performed by the CSPs, so customers' IT teams can spend time on achieving more important business goals. And, because CSPs scale across thousands of customers, they develop and deploy automated modern tools for these management activities.

Another aspect of improved productivity for customers is a result of velocity; since the required computing resources can be provisioned and deployed almost instantly, the customer can begin prototyping immediately.

In addition to velocity, cloud computing has various differing levels of services: from IaaS to SaaS. The customer may choose what they want to focus on and leverage rest as a service from the CSP. All in all, cloud computing brings productivity across the board.

Performance

The most prominent cloud service providers are deployed worldwide. They apply secure and fast networks and apply the latest technologies to secure their data and upgrade the hardware resources (compute, storage, memory, etc.) regularly with the latest generation of fast and efficient computing hardware. These attributes make best-in-class performance available to cloud service customers all around the world, reducing latency for geo-dispersed customers by means of colocating the cloud resources and customer in the same geography.

Ease of Use and Maintenance

Cloud service providers offer several tools, technologies, and controls to strengthen the security and protection of data and apps. Additionally, the cloud service providers keep the security patches, features, and tools up to date, which results in improved security.

Combined with other benefits, cloud computing makes software development, deployment, and maintenance easy, hassle-free, and secure while being economical.

Cloud Deployment Configurations

So far, we've talked about what cloud computing is in general and the benefits it brings. When it comes to deploying to the cloud, there are many ways to implement that. There are many different decisions that could impact the implementation and the deployment, which makes one instance of cloud deployment look very different from another. Some such decisions include whom the cloud is accessible to, where it is located and hosted, how the security is implemented, and so on.

Broadly, there are three different ways to deploy cloud services: private cloud, public cloud, and a mix of the two called hybrid cloud. In the following sections, we will talk about each of them and their related trade-offs.

Private Cloud

A private cloud refers to a setup where the cloud computing resources are designed and used exclusively by a single organization. The private cloud usually resides behind a firewall and on a private network. A completely on-premises private cloud can be physically located on the on-site data center. The organization may host and manage the private cloud on their own. However, some organizations hire third-party service providers to host their private cloud.

Private cloud solutions offer both security and control. The benefits, however, come at a cost. The organizations that own the cloud are responsible for the creation, deployment, and management of all the infrastructure and software, which makes the private cloud a less economical model than the public cloud in most of the cases. The private cloud could still make sense for the businesses with very stringent regulatory requirements.

Public Cloud

As the name suggests, public clouds are owned and operated by third-party service providers, known as cloud service providers (CSPs). These cloud service providers deliver computing resources over the Internet for their subscribers. Amazon, Microsoft, and Google are some examples of public cloud service providers. These cloud service providers specialize in the business and own and manage all hardware, software, and other supporting infrastructure. The customers of cloud service providers subscribe to and access these services over the Internet. Because of the sharing of cloud resources across the customers, public cloud offerings may be more economical for the majority of customers and use cases. The public cloud model provides smaller organizations the benefits of scale and economy.

Hybrid Cloud

A hybrid cloud, as one can guess, combines public and private clouds. The hybrid cloud is designed to allow two platforms to work together, seamlessly. The hybrid cloud model brings the best of both worlds (private cloud and public cloud) together: provide the scalable computing power of a public cloud with the security and control of a private cloud.

Ideal Cloud Deployment Configuration

As we discussed in the preceding sections, there is a trade-off between the public and private cloud deployment configurations. For example, the private cloud configuration may give you control and may be better equipped to store sensitive information regarding the corporation. The public deployment may provide better flexibility and scale. The performance and uptime may be better for public cloud deployment because the public cloud service providers specialize in that.

The use case itself may define whether a public or private cloud deployment is more suitable. For instance, there may be scope for optimization in terms of data going into (ingress)/and going out of (egress) the private cloud setups on-premises, while with public setups, there could be potentially increased cost for such data movement.

Because of these reasons, many larger organizations combine the benefits of the two and use a hybrid cloud deployment. There is no standard guidance on what mix (private vs. public cloud) is the ideal. Organizations need to carefully evaluate and come up with the ideal setup based on their usages and trade-offs. In some cases, it could even require a multi-cloud model.

Multi-cloud Model

In cases where a single public cloud isn't enough to meet an organization's computing needs, they may have to use services from multiple public cloud service providers and deploy a little more complex hybrid cloud architecture that combines a private cloud with multiple public cloud services. This model of deployment is known as multi-cloud. While a hybrid cloud consists of a public and a private cloud, a multi-cloud environment is a lot more complex and engages multiple public cloud service providers.

Cloud Configuration Interface/Mechanism

In the earlier sections, we talked in detail about cloud services, the complex deployment models, benefits, and so on. Although all of that might seem very fancy, at the most basic level, accessing cloud services is not very different from accessing a remote system over a network. On top of that, the CSPs may provide more user-friendly ways to access and manage the services.

However, before we can access anything, we need to subscribe to the cloud services offered by the CSP. Once you have the subscription, the CSP will provide a user interface to create a logical custom machine by assembling computing resources like CPU, memory, storage, and so on. Once the machine is ready and the network address is allocated, it's pretty much like accessing any remote system over a network.

Note To lure customers, CSPs may provide free access for some limited usage and time period.

At a high level, cloud computing is made of two components: front end and back end. The front end enables the customers to subscribe to, manage, and access the cloud services provided by the cloud service provider. The back end is the actual cloud, and it is made of the resources (compute, memory, storage, network, etc.) that are required to enable and support cloud computing services.

Figure 7-4 provides a logical view for interfacing with cloud services and resources.

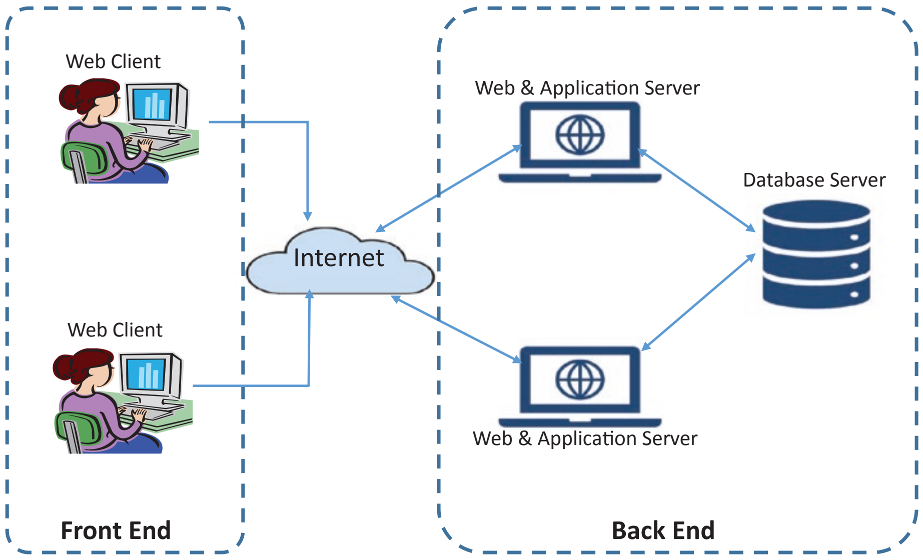

Figure 7-4. Logical View of Interfacing with Cloud

Note Logically, using cloud services is similar to accessing a remote machine over a network.

Cloud Service Providers

The following are several cloud service providers (CSPs) in the market today. This list is not meant to be comprehensive and by no means in any specific priority order:

- Google Cloud
- Microsoft Azure
- Amazon AWS
- Alibaba Cloud
- IBM
- Oracle
- And many more, like Linode, VMware, and so on

Considerations in Choosing a CSP

As we saw in the preceding section, there are a variety of cloud service providers. Although most of them provide similar capabilities and services, most CSPs offer proprietary and nonstandard (e.g., open source) services, supported by proprietary architectures and technologies. Choosing to use these proprietary services and technologies and developing solutions using them might lead to customers getting locked in with the CSP. The lock-in could happen due to various reasons including

- Using custom CSP solutions, services, and mechanisms
- Defining architectural decisions based on specific services by the CSP
- Designing and developing with the specific CSP in consideration

The resulting lock-in could lead to several negative side effects, for instance:

- Significant cost, time, and effort during migration
- Potential downtime during migration
- Difficulty migrating to a different, lower-cost CSP in the future

With various cloud computing models, lock-in risk levels are different across IaaS, SaaS, and PaaS. For example, let's talk about PaaS. A specific PaaS platform from a CSP may support only limited and proprietary technologies, for example, specific web frameworks, languages, databases, and so on. This can lead to development of application architecture that is tied to the technologies offered by that specific CSP resulting in the application getting locked in the CSP. Again, across IaaS, PaaS, and SaaS, the lock-in risk is lowest in the IaaS model because IaaS functionality is broadly similar and there are several standards covering them.

However, one may ask why lock-in is a bad thing. Why would one need to migrate from one CSP to another? The argument could be that while choosing a CSP for the very first time, we perform due diligence to choose the best CSP for the specific need and use case. Once we do that, what is the need of ever considering migrating? In the following section, we will discuss what causes customers to move from one CSP to other.

Motivation for Switching CSPs

It is evident that cloud service customers (CSCs) evaluate CSPs at the time of choosing and make the best choice based on their particular use case and data available. However, requirements can change quickly, and that can be motivation for moving from one CSP to another. Some of the reasons that could cause the need for migration are covered in the following sections. These reasons are not meant to be comprehensive, but simply some of the most common ones.

Usage and Pricing Change

One of the primary reasons for a customer moving from one CSP to other relates to a usage and/or pricing change.

The CSPs have a pricing model, and when a customer evaluates the CSPs for pricing, they use a model for usage levels and patterns. However, actual usage could turn out to be very different than the original model, and that could make the original evaluation very far from reality. It may turn out that for the actual usage model, a different CSP may be more economical.

Also, usage could evolve and change because of the services deployed on the cloud becoming more or less popular than originally anticipated.

The other potential reason is that the CSPs revise their fees from time to time, and based on these revisions, the current CSP might become less attractive than some other.

CSP Ecosystem Change

The other category of changes that could lead to a customer moving from one CSP to another relates to changes in the CSP ecosystem itself. Some of those include

- CSPs Moving In and Out of Business: Although rare for some of the big players in the CSP business, some of the smaller ones may move out of business or change hands. Also, like the CSPs moving out of business, there could be a scenario where a new CSP emerges and may have better offerings in terms of pricing, scalability, and support.
- CSP Abandoning a Specific Service: The overall cloud computing services and offerings are evolving as we speak, and as a side effect of that, the CSPs may abandon or change specific proprietary offerings.

Regulatory, Privacy, and Business Dynamics Change

Another category that could motivate changing CSPs is related to regulatory, privacy, and business dynamics changes:

- Rules, Regulations, and Policy: There could be a government regulation or other policy changes that could dictate hosting of certain services in certain geographies or a specific way that in turn results in need for moving from one CSP to another.
- Discovery of a New Vulnerability/Loophole or Limitation at a CSP: While in deployment, the customer could realize/discover a new vulnerability, loophole, or limitation that would require the customer to move to another CSP.
- Business Dynamics Change: The business environment is fluid, and things change quickly. So, for instance, the CSP and CSC could move into the same business, and then there could be some conflict of interest in hosting/supporting the data from competition.

Considerations for Developing Portable and Interoperable Cloud Solutions

As we've discussed in earlier sections, it's evident that cloud service customers (CSCs) may need to move from one CSP to another. That directly implies that one must develop solutions in a way that they are portable and interoperable across CSPs. In the following sections, we will talk about what we mean by portability and interoperability, as well as mechanisms we can apply to achieve this.

Interoperability vs. Portability

Generally speaking, interoperability can be thought as the measurement of the degree to which diverse systems or components can work together successfully.

In cloud computing context, there are two parties: cloud service customer and cloud service provider. They both interact over a network connection using a prescribed interface or API. There are different aspects of the cloud service, and there are typically separate interfaces for dealing with these different aspects of the cloud service.

For instance, there could be functional interfaces of the cloud service itself, interfaces for authentication and authorization, interfaces for administration of the cloud services, and even more interfaces relating to such business aspects as billing and invoicing. The objective of interoperability is that these interfaces are standardized in a way that they are interoperable so that there is minimal impact to the CSC's component while moving from one CSP to other.

Similarly, portability refers to the ability of a customer to move and adapt their applications (application portability) and data (data portability)

- Between a customer system and CSP
- From one CSP to another CSP

Portability is important because the lack of it leads to considerable cost and effort when you do have to change systems. There are two aspects of portability that need consideration:

- **Cloud data portability** is the ability to easily transfer data: from one CSP to another and between a CSP and customer's system.

- **Cloud application portability** is the ability to easily transfer an application or application component: from one CSP to another and between a CSP and customer's system. This typically only applies to IaaS and PaaS services, since in the case of a SaaS service, the application belongs to the cloud service provider and there is no use case of that being ported elsewhere by the customer.

As per the *Cloud Standards Customer Council's Interoperability and Portability for Cloud Computing: A Guide Version 2.0* (2017), the interoperability and portability aspects of the cloud solution could pictorially be depicted as in Figure 7-5.

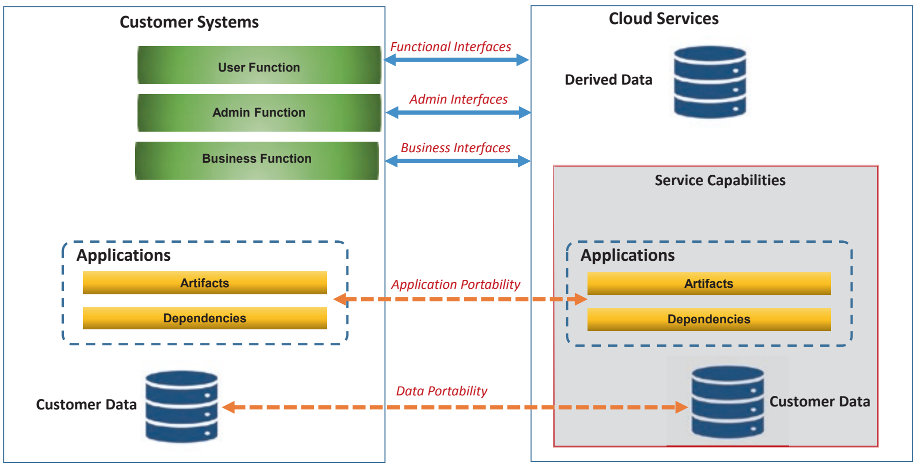

***Figure 7-5.** Elements of Interoperability and Portability in Cloud Services*

In Figure 7-5, the first three interfaces relate to the interoperability aspect, while the rest relate to the portability aspect of the cloud services.

Interoperability Challenges

There are several reasons why interoperability challenges exist across CSPs. The key reason is interfaces and services offered by CSPs are not standardized. Different cloud service providers use different APIs even for comparable cloud services. The greatest level of interoperability is made available by IaaS, while the PaaS cloud services have lower levels of interoperability because there are few interface standards for PaaS. Recently, open source platforms such as Cloud Foundry are gaining momentum and provide common open source–based solutions that can run on any CSP platform. Similarly, SaaS applications present the greatest interoperability challenge. That is because there are very few standards for SaaS applications. Therefore, moving from one SaaS application to another SaaS application usually requires interface changes.

To mitigate these challenges, the cloud service customer usually has an isolation or mapping layer between their own applications and the cloud service interface. For instance, technologies such as enterprise service buses (ESBs) can be used to build these isolation layers. The other potential mitigation option is to use the services offered by an inter-cloud provider (aka cloud service broker), who maps a “standard” interface offered to the customer to a varying set of interfaces offered by several different CSPs.

Portability Challenges

As discussed earlier, the portability challenges are different across IaaS, PaaS, and SaaS. The biggest challenges are for applications built for PaaS platforms, because

- Platforms can vary widely between different CSPs.
- The app environment can differ substantially across CSPs.

For example, to be scalable and elastic, a PaaS platform may enforce a specific way to manage data that may not be supported by other PaaS platforms. Although there are some standards relating to PaaS that are picking up momentum, for IaaS cloud services, there are several standards that are already in practice. Using these standards results in improved portability of applications.

To minimize the portability challenges, there are a couple of best-known strategies in place: one is increasing adoption of common open source PaaS platforms such as Cloud Foundry, and another is leveraging containerization that allows independent deployment of applications. In the following sections, we discuss containerization and orchestration and how they help enable portability.

Containers, Docker, and Kubernetes

In order to make software and applications portable, the first consideration is how we can isolate the application so that it has limited dependency on and expectation from the underlying environment it is going to be deployed on. That is where the idea of containers originated and was first implemented by Docker. Essentially, a container is a standard unit of software that packages up an application and all its dependencies so that the application runs the same, regardless of the computing environment (assuming similar compute resources are available).

Containers allow a developer to package up an application with all the parts it needs, such as libraries and other dependencies, and deploy it as one package: a container image. Container images become containers at runtime.

So, in a way, containers are like virtual machines (VMs) in terms of isolation benefits; however, they work at a slightly different level: containers virtualize the operating system, while virtual machines virtualize the hardware. As a by-product of this difference, containers are lighter weight and efficient as compared to virtual machines.

Note A container is a packaging of an application and its dependencies as a self-contained package, and Docker is one implementation of this concept. There is an initiative to standardize the container format for wider portability called the *Open Container Initiative*.

Figure 7-6 shows the similarity and the difference between virtual machine– and container (Docker)-based deployments.

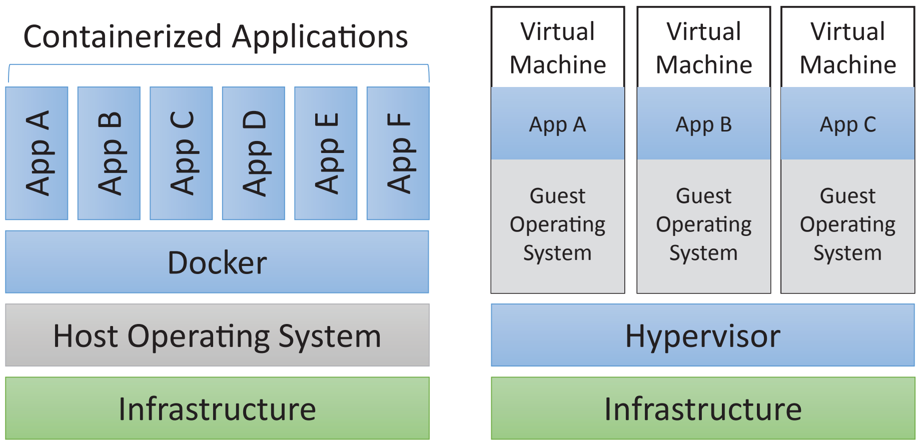

Figure 7-6. *Containers vs. Virtual Machines*

As is evident from Figure 7-6, containers are an abstraction above the OS layer that packages an application and its dependencies together. Multiple containers can run on the same machine (virtual or physical) and share the host OS kernel with each other. Each container runs as isolated processes in user space. Containers are lighter weight in terms of space and computing overhead than VMs and therefore more efficient than VMs.

Virtual machines have been around for some time now and are well known. They provide abstraction of physical hardware each running its own copy of an operating system and the applications, necessary binaries, and libraries. One or more VMs can run on the supplied hardware, but each VM has a full OS.

As such, VMs are much heavier than containers. However, they are not mutually exclusive. In fact, in today's world, both virtual machines and containers are combined to leverage the best of both. So, in practice, there could be one capable machine running multiple virtual machines, each of which in turn running more than one container as needed.

The use of containers requires three additional capabilities or tools:

- Builder: We need the tools and technology to build a container image.
- Engine: We need the tools and technology to run an image. A running image is a container.
- Orchestration: We need the tools and technology to effectively manage container instances.

The first two pieces – Builder and Engine – are very clear. We need Builder to create the image and Engine to run that image. Docker serves that purpose. Essentially, Docker is a set of command line tools and runtime to create and run the container images.

Figure 7-7 demonstrates a high-level architecture of Docker. The command line tools like "Docker build" and "Docker run" are the client. These commands are used by the end users to create and run Docker images. The Docker images created are registered to a global registry that allows an image to be built once and then used multiple times. The "Docker daemon" creates and runs the images in contained environments (containers).

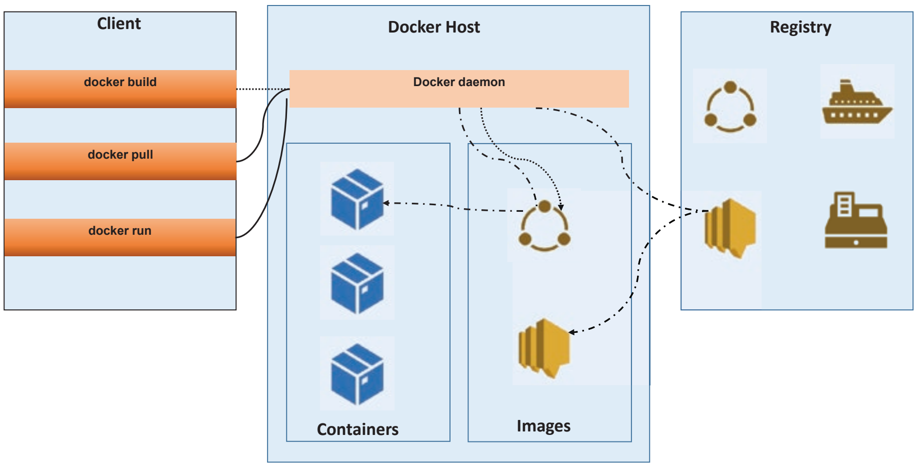

Figure 7-7. *Docker Architecture*

Containers are a portable way to bundle and run applications. However, in a production environment, we need to manage the containers that run and ensure that there is no downtime. Additionally, we need to be able to scale the number of containers based on demand. Kubernetes (K8s) does exactly that. It standardizes the process of deploying and managing the sets of containers that are required for a given application.

Note Kubernetes is a container orchestration framework that manages deployment of containers to provide resiliency to the system.

Figure 7-8 shows the high-level Kubernetes architecture. At the top layer, there is the command line interface (CLI), “kubectl,” that works with the control plane to orchestrate the deployment. Kubernetes nodes manage and run pods. The nodes could either be virtual or physical machines. A node can host and manage one or more pods. Each of these pods can run one or more containers. From the Kubernetes perspective, pods are the unit of deployment. There could be pods with just one container; however, for more complex deployments, there is likely to be more than one container to a pod. “kubelet” monitors and manages containers within a pod. All these pieces put together is called a Kubernetes “cluster.”

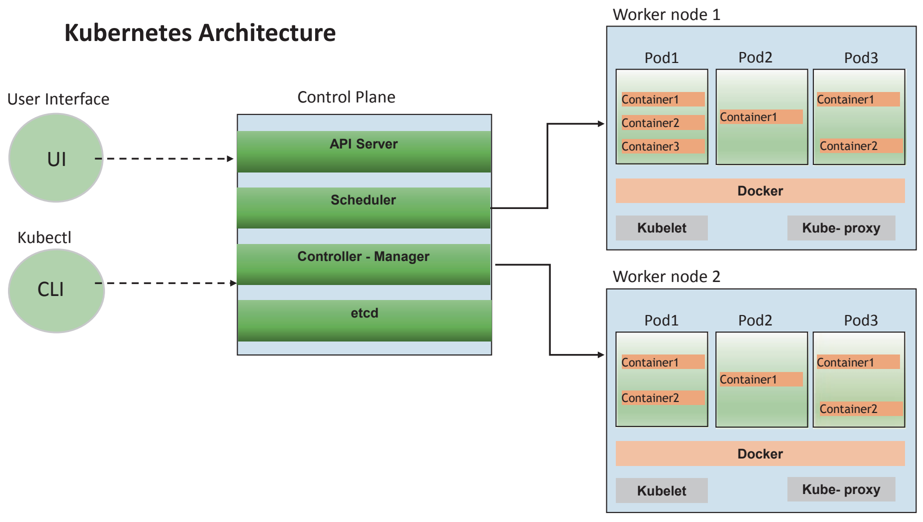

Figure 7-8. *Kubernetes architecture*

Benefits of Containers and Container Orchestration

Although we talked about containers, Docker, and Kubernetes in the context of portability, which clearly is one of the most important benefits of using containers, there are other benefits. The key ones are described next.

Security

First and foremost, it adds extra security through isolation. While the isolation and the security can be architected by the virtual machines as well, however, since containers include only the necessary code and libraries needed by the application, they have less code to be compromised, compared to VMs.

Scalability

Applying microservices, containers, and container orchestration together, we can automatically scale up and down the application resources based on the demand. For illustration, during the festive season when the demand and load on the ecommerce portal, say Amazon, grows, more instances of microservice-based containers can be automatically created by Kubernetes to bear the load.

The Way Forward

The cloud movement is still relatively young by technology standards and is still rapidly evolving. How this evolution will impact the ecosystem is anyone's guess. That said, with faster and faster networks and throughput, we are headed to a world where everything goes into/through the cloud. All the applications and services reside in the cloud (public, private, and/or hybrid). For example, one has moved to the cloud-based "Office 365." This model enables use of SaaS and pay-per-use.

There are several other factors in play while defining the future of clients and clouds. First and foremost, even though there is wider availability of the Internet across the globe, there still is a large portion of the population that is not yet well connected with the Internet. Even for the population where there is reasonable Internet access, bandwidth and throughput are not always great enough to make it a seamless experience.

Additionally, there are new uses that require real-time responses. For a real-time response, going back and forth between the cloud and client over a network may not be performant, so clients still need reasonable compute power. For instance, in gaming and other interactive applications, the experience would be compromised if we fully relied on the cloud for all the processing.

Finally, data protection and privacy concerns could prevent users from doing everything in the cloud.

So, in summary, the smart clients are not going away anytime soon. In fact, they are going to co-exist and continue their symbiotic relationship with the cloud.

Leading organizations have already realized they need both robust client and cloud solutions working together and have started creating architectures to provide the best experience for users. Specifically, the idea is to look at the ways we can leverage the capabilities of both client and cloud in developing a solution and offload the services at the client or cloud, respectively, whichever is more efficient for the job.

Recommendations

Based on the discussion on portability and interoperability, we would like to consolidate the key recommendations for cloud service customers, as follows:

- Portability and interoperability should be key considerations when choosing the CSP. Also, the portability and interoperability requirements should be part of the agreement with the CSP.
- Use open and standard technologies, protocols, and mechanisms, and avoid using CSP proprietary solutions, where practical. Choose a CSP that supports these open and standard technologies.
- Ensure applications follow service-orientated architecture (SOA) and employ standard APIs for interoperability. Additionally, use protocol adapters like enterprise service buses for handling protocol mismatches.
- Leverage containers for virtualizing applications and artifacts to ensure portability.

Summary

In this chapter, we presented the fundamentals of cloud computing, its benefits, and the various potential deployment configurations and why choose one over another. We introduced a few of the cloud service providers (CSPs) in business today, as well as the key considerations in choosing a cloud service provider for a specific use case or organization. We emphasized the need for portability and interoperability as first-order criteria to avoid lock-in to a specific CSP. We also covered how to develop portable and interoperable solutions before closing with a brief conversation on how the client and cloud will potentially evolve in the future.

References and Further Reading

- *European Journal of ePractice*. Three Dimensions of Organizational Interoperability: www.epractice.eu/files/6.1.pdf
- Cloud Standards Customer Council (2017). Practical Guide to Cloud Computing: www.cloud-council.org/deliverables/practical-guide-to-cloud-computing.htm
- Cloud Standards Customer Council (2015). Practical Guide to Cloud Service Level Agreements: www.cloud-council.org/deliverables/practical-guide-to-cloud-service-agreements.htm
- Cloud Standards Customer Council (2016). Public Cloud Service Agreements: What to Expect and What to Negotiate*:* www.cloud-council.org/deliverables/public-cloud-service-agreements-what-to-expect-and-whatto-negotiate.htm

- Cloud Standards Customer Council (2016). Practical Guide to Hybrid Cloud Computing: www.cloud-council.org/deliverables/practical-guide-to-hybrid-cloud-computing.htm
- ISO/IEC 19941 Cloud Computing – Interoperability and Portability: www.iso.org/standard/66639.html
- Cloud Standards Customer Council (2017). Practical Guide to Cloud Management Platforms: www.cloud-council.org/deliverables/practical-guide-to-cloud-management-platforms.htm
- Cloud Standards Customer Council (2013). Migrating Applications to Public Cloud Services: Roadmap for Success: www.cloud-council.org/deliverables/migrating-applications-to-public-cloud-services-roadmapfor-success.htm
- Production-Grade Container Orchestration: https://kubernetes.io/
- Containers and Dockers: www.docker.com/resources/what-container
- *Open Container Initiative:* www.opencontainers.org/
- *Open Virtualization Format:* www.dmtf.org/standards/ovf
- Cloud Foundry: *Cloud Application Platform:* www.cloudfoundry.org/
- *Cloud Data Management Interface (CDMI):* www.snia.org/cdmi