Unit II

1. Business Case for Advanced Cloud Computing

The business case should calculate the costs of migrating to the cloud -- which include the cost of moving systems over, as well as the cost of running services in the cloud after migration -- and then compare them to the costs of keeping systems in-house.

At a recent Amazon Web Services (AWS) event in London the company set out how it builds a business case for customers who are looking at moving services from the traditional on-premise model, where they keep their data and applications in their own data centers, to having them hosted in the cloud, and what companies need to take into account.

According to AWS, the business case has to start with the business objectives: why the organisation is intending to move to the cloud, and what it wants to get out of it. This is followed by the discovery process: understanding the existing infrastructure and associated costs.

Depending on the customer, gathering and analysing that data can range from a one- or two-day project all the way up to detailed projects lasting several weeks.

Building a business case isn't only needed for a full-scale move to the cloud. It's also worth doing by companies using the cloud in different ways, starting with standalone cloud projects, as well as when companies are testing out a hybrid model.

Infrastructure business case

The infrastructure savings can be the most significant part of the business case in terms of cost savings.

Infrastructure costs that need to be factored in can include the cost of facilities like data centers, including the cost of lease time remaining and any penalties. The cost implications of an incomplete move to the cloud also needs to be considered: if some apps cannot be moved to the cloud and that means the infrastructure can't be reduced, or the data center cannot be shuttered, then those apps become a lot more expensive to run on their own.

Other factors to include are the cost of connectivity, such as leased lines. An obvious key consideration is the number of physical servers, virtual servers, and details of specifications like CPUs, cores and RAM. The cost of storage including SAN, NAS, and direct attached storage also needs to be added, as well as any potential upcoming hardware refresh plans that could potentially be avoided.

Depreciation and amortisation, plus data center management costs and the actual server utilisation needs to be added into the model too.

"Looking at the server usage pattern is also key to getting the right sizing,"Other things to factor in include cooling, backup, compliance and certification, plus end-of-life and decommissioning costs, he added.

Applications business case

For applications there are a number of options ranging from leaving them exactly where they are, retiring them permanently, re-hosting them in the cloud unchanged, moving them to cloud with some changes, completely rebuilding them for the cloud or buying an entirely new software-as-a-service package. Each of these will have different cost implications.

Under application costs, issues to consider include:

- The number of workloads, and how they map to physical servers
- The cost of application changes, application maintenance costs, regulation and compliance
- Licence transferability and upcoming licence renewals
- Application requirements such as any SLAs, disaster recovery, security and access requirements

Staffing business case

The cloud business case also needs to include people costs, which can be second only to the infrastructure costs in some cases. Companies considering a move to the cloud need to consider, for both staff and contractors, the cost of recruitment, retention, replacement and retirement, training and development, plus the physical space requirement (e.g. offices) plus equipment and any other services. Companies also need to consider any other third-party costs and contracts -- including early termination policies.

It's also important to take into account more nebulous factors. Cloud advocates will argue that using hosted services allows companies to move faster -- for example by updating applications faster than was previously possible. But it's also worth bearing in mind potential cloud drawbacks, including the risk of being locked into one vendor for the majority of your tech infrastructure.

2. Advanced Computer System

Advanced computing is a broad term used to describe either a specific type of high-end computer and the processes undertaken on it, or a set of skills used on personal computers. Both meanings are quite different, and there is no strict definition of the phrase, so that what one person means when they say advanced computing might be very different than what another person says. Generally, if a course is offered in this subject it is referring to advanced computer skills, while an agency that promotes advanced computing is likely talking about high-end computing.

Elements of Advanced Computing Systems

Operating advanced computer systems entails working with computer architecture, hardware and software and computer systems. Computer architecture provides a high performance (HPC) potential, while hardware and software is used for high performance graphics. Systems are able to hold a large amount of data and store vast quantities of information. A computer needs the capability to provide networking resources and advanced software.

A computer's infrastructure, combined with an operating system (OS), embeds and distributes information throughout a computer's architecture. This ensures the elements work in conjunction with one another to provide the capability of a computer to operate effectively.

3. Advanced Cloud Computing Modern Architecture and Framework

Cloud computing architecture is simple; it clearly states the components and subcomponents embedded in it There's no question that cloud computing is here to stay. It touches every part of our lives today, offering many advantages in terms of flexibility, storage, sharing, maintenance, and much more.

A standard internet connection or a virtual network provides us access to cloud-based applications and services like Google Docs, Skype, and Netflix. Most companies are shifting their businesses into the cloud as they require significant storage, which cloud platforms provide. A cloud computing architecture provides higher bandwidth to its users due to which data over the cloud can be used from anywhere across the world at any time. Due to its architecture, it not only shares resources among client source consumers but also with open source communities like Microsoft and Red hat.

What is Cloud Computing?

Cloud computing refers to services like storage, databases, software, analytics, and other platforms that are accessible via the internet. It is any service that can be delivered without being physically close to the hardware. For example, Netflix uses cloud computing for its video streaming services. Another example is G Suite, which runs entirely on the cloud.

Simply put, Cloud Computing refers to the delivery of on-demand resources (such as a server, database, software, etc.) over the internet. It also gives the ability to build, design, and manage applications on the cloud platform.

Note: Companies offering these computing services are referred to as cloud providers.

Cloud Computing Service Providers

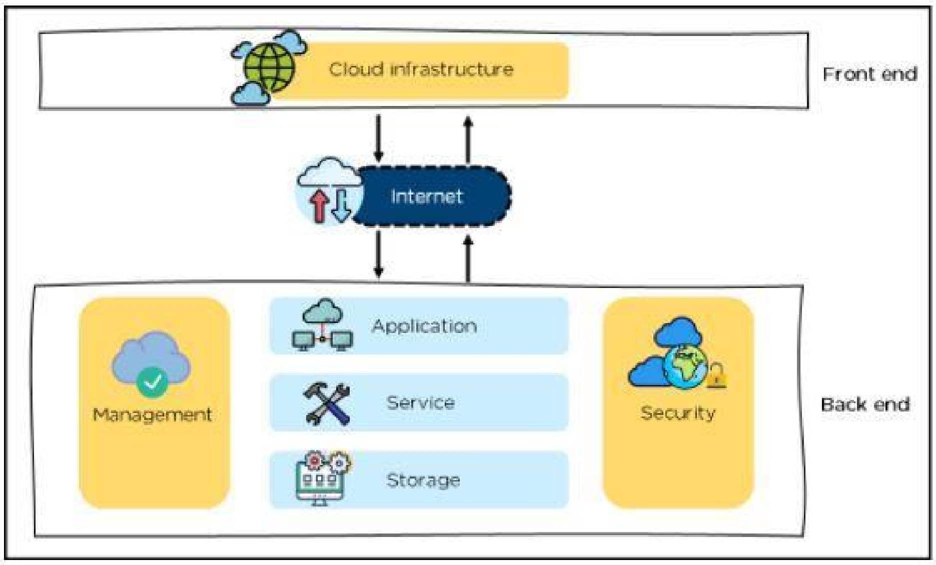
A few of the most popular cloud computing service providers include:

- Microsoft Azure
- Amazon Web Services (AWS)
- Google Cloud
- Alibaba Cloud
- IBM Cloud
- Oracle
- Salesforce
- SAP
- Rackspace Cloud
- VMWare

As one of the top three cloud providers available, there are plenty of career opportunities related to GCP. Simplilearn's Google cloud certification provides you with the foundation you will need to start or enhance your current career working with this comprehensive cloud platform. Get started today!

Cloud Computing Architecture

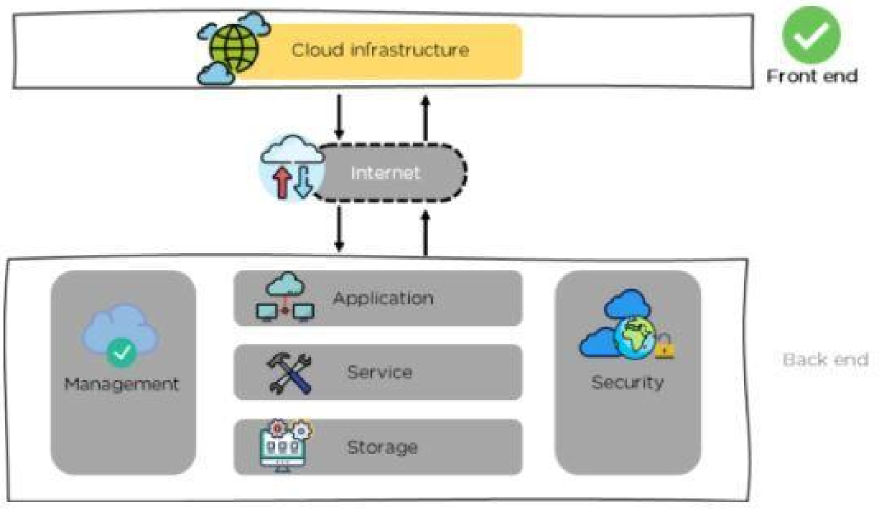
Cloud Computing Architecture is divided into two parts, i.e., front-end and back-end. Front-end and back-end communicate via a network or internet. A diagrammatic representation of cloud computing architecture is shown below:



Cloud Computing Architecture

Front-End

- It provides applications and the interfaces that are required for the cloud-based service.
- It consists of client's side applications, which are web browsers such as Google Chrome and Internet Explorer.
- Cloud infrastructure is the only component of the front-end. Let's understand it in detail.

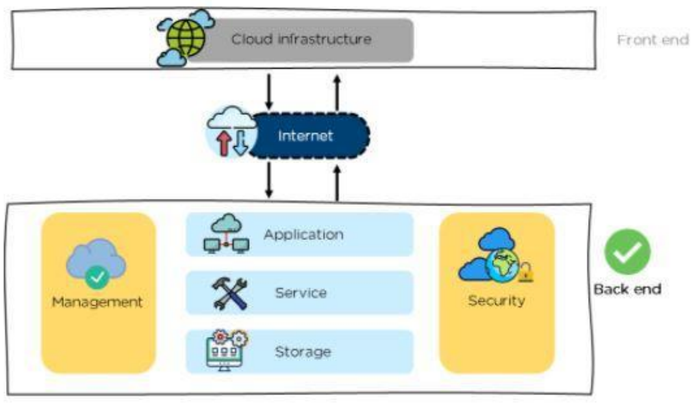


Front-end - Cloud Computing Architecture

- Cloud infrastructure consists of hardware and software components such as data storage, server, virtualization software, etc.
- It also provides a Graphical User Interface to the end-users to perform respective tasks.

Back-End

It is responsible for monitoring all the programs that run the application on the front-end
It has a large number of data storage systems and servers. The back-end is an important and huge part of the whole cloud computing architecture, as shown below:



Back-end - Cloud Computing Architecture

The components of the back-end cloud architecture are mentioned below. Let's understand them in detail one by one.

Application

- It can either be a software or a platform
- Depending upon the client requirement, the application provides the result to the end-user (with resources) in the back end

Service

- Service is an essential component in cloud architecture
- Its responsibility is to provide utility in the architecture

- In a Cloud, few widely used services among the end-users are storage application development environments and web services

Storage

- It stores and maintains data like files, videos, documents, etc. over the internet
- Some of the popular examples of storage services are below:
    - Amazon S3
    - Oracle Cloud-Storage
    - Microsoft Azure Storage
- Its capacity varies depending upon the service providers available in the market

Management

- Its task is to allot specific resources to a specific task, it simultaneously performs various functions of the cloud environment
- It helps in the management of components like application, task, service, security, data storage, and cloud infrastructure
- In simple terms, it establishes coordination among the cloud resources

Security

- Security is an integral part of back-end cloud infrastructure
- It provides secure cloud resources, systems, files, and infrastructure to end-users
- Also, it implements security management to the cloud server with virtual firewalls which results in preventing data loss

Benefits of Cloud Computing Architecture

The cloud computing architecture is designed in such a way that:

- It solves latency issues and improves data processing requirements
- It reduces IT operating costs and gives good accessibility to access data and digital tools
- It helps businesses to easily scale up and scale down their cloud resources
- It has a flexibility feature which gives businesses a competitive advantage

- It results in better disaster recovery and provides high security
- It automatically updates its services
- It encourages remote working and promotes team collaboration

Going ahead, let’s have a look at the components of cloud computing architecture.

Cloud Computing Architecture Components

Some of the important components of Cloud Computing architecture that we will be looking into are as follows:

- Hypervisor
- Management Software
- Deployment Software
- Network
- Cloud Server
- Cloud Storage

Hypervisor

- It is a virtual machine monitor which provides Virtual Operating Platforms to every user
- It also manages guest operating systems in the cloud
- It runs a separate virtual machine on the back end which consists of software and hardware
- Its main objective is to divide and allocate resources

Management Software

- Its responsibility is to manage and monitor cloud operations with various strategies to increase the performance of the cloud
- Some of the operations performed by the management software are:
    - compliance auditing
    - management of overseeing disaster
    - contingency plans

Deployment Software

- It consists of all the mandatory installations and configurations required to run a cloud service
- Every deployment of cloud services are performed using a deployment software
- The three different models which can be deployed are the following:

- SaaS - Software as a service hosts and manages applications of the end-user.

Example: Gmail

Image_Name: PaaS

- PaaS - Platform as a service helps developers to build, create, and manage applications.

Example: Microsoft Azure

- IaaS - Infrastructure as a service provides services on a pay-as-you-go pricing model.

Network

- It connects the front-end and back-end. Also, allows every user to access cloud resources
- It helps users to connect and customize the route and protocol
- It is a virtual server which is hosted on the cloud computing platform
- It is highly flexible, secure, and cost-effective

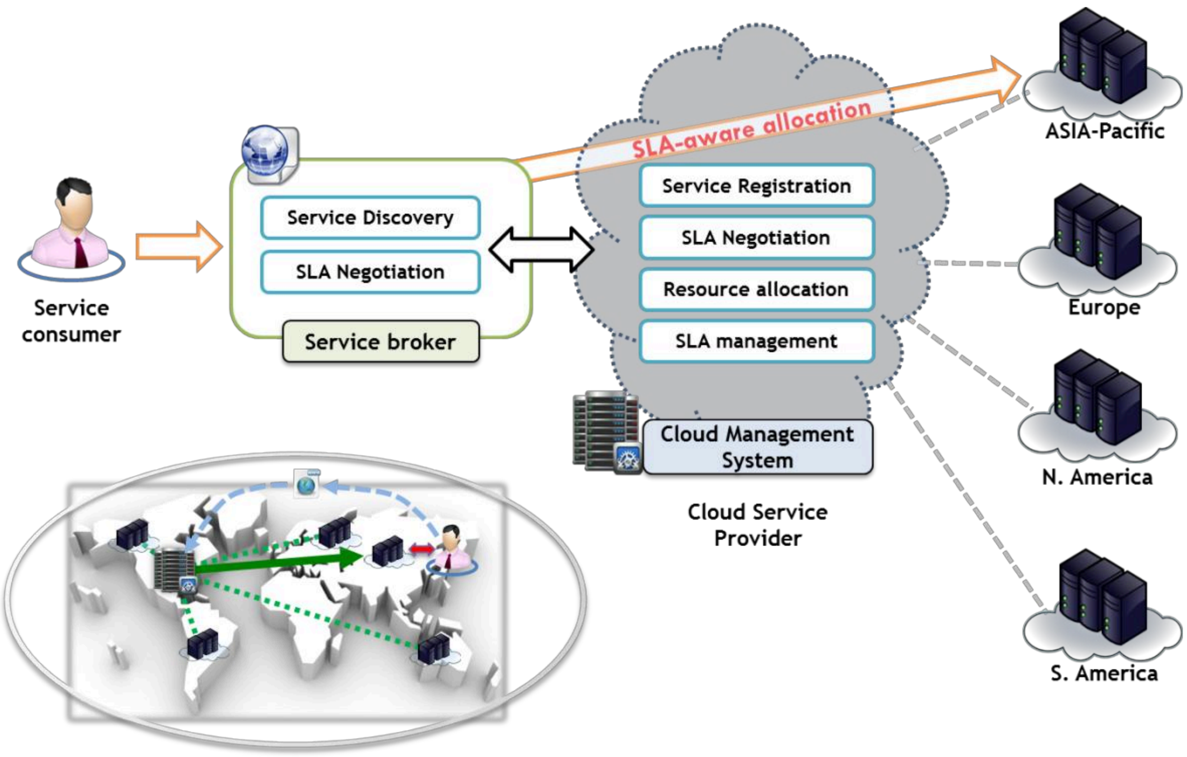
Cloud Storage

- Here, every bit of data is stored and accessed by a user from anywhere over the internet
- It is scalable at run-time and is automatically accessed
- Data can be modified and retrieved from cloud storage over the web

Conclusion

Cloud computing architecture gives an environment where organizations can securely build applications and use cloud services based on the client requirement. So, with this, we got a complete run-down on what Cloud Computing Architecture is. In this article, we learned what cloud computing is, the benefits of Cloud Computing architecture, the architecture of cloud computing, and components of cloud computing architecture.

3.b. Cloud Computing Framework



Identified ten major categories or patterns of cloud computing technology:

1. Storage-as-a-Service
2. Database-as-a-Service
3. Information-as-a-Service
4. Process-as-a-Service
5. Application-as-a-Service
6. Platform-as-a-Service
7. Integration-as-a-Service
8. Security-as-a-Service
9. Management/Governance-as-a-Service
10. Testing-as-a-Service

1. **Storage-as-a-service -** as expected, is the ability to leverage storage that physically exists remotely, but is logically a local storage resource to any application that requires storage. This is the most primitive component of cloud computing, and is a component or pattern that's leveraged by most of the other cloud computing components.

   Storage-as-a-service providers include Amazon S3, Box.net, and Google Base.

2. **Database-as-a-service** provides the ability to leverage the services of a remotely hosted database, sharing it with other users, and having it logically function as if the database were local. Different models are offered by different providers, but the power is to leverage database technology that would typically cost thousands of dollars in hardware and software licenses. **Database-as-a-service** providers include Amazon SimpleDB, Trackvia, and Microsoft SSDS.
3. **Information-as-a-service** refers to the ability to consume any type of information, remotely hosted, through a well-defined interface such as an API, for example, stock price information, address validation, credit reporting, etc. There are over a 1,000 sources of information that can be found these days, most of them listed in www.programmableweb.com
4. **Process-as-a-service** refers to a remote resource that's able to bind many resources together, either hosted within the same cloud computing resource or remote, to create business processes. These processes are typically easier to change than applications, and thus provide agility to those who leverage these process engines that are delivered on-demand. **Process-as-a-service** providers include Appian Anywhere, Akemma, and Intensil.
5. **Application-as-a-service**, also known as **software-as-a-service** (**SaaS**), is any application delivered over the platform of the Web to an end user, typically leveraging the application through a browser. While many associate application-as-a-service with enterprise applications, such as Salesforce SFA, office automation applications are indeed. **applications-as-a-service** as well, including Google Docs, Gmail, and Google Calender. This was really the first drive into modern cloud computing, but is based on the more traditional timesharing model from years past where many users shared one application and one computer.

   Application-as-a-service providers include Salesforce, Netsuite, Oracle On Demand, and Google Apps.
6. **Platform-as-a-service** is a complete platform, including application development, interface development, database development, storage, testing, etc., delivered through a remotely hosted platform to subscribers. Based upon the traditional timesharing model, modern platform-as-service providers provide the ability to create enterprise-class applications for use locally or on-demand for a small subscription price or for free.

Platform-as-a-service providers include Bungee Labs Connect, Coghead, Google App Engine, Long.jump, Force.com, Etelos, Oracle SaaS, and Apprenda SaaSGrind.

7. **Integration-as-a-service**, is the ability to deliver a complete integration stack from the cloud, including interfacing with applications, semantic mediation, flow control, integration design, etc. In essence, integration-as-a-service includes most of the features and functions found within traditional EAI technology, but delivered as a service.
   Integration-as-a-service providers include Amazon SQS, OpSource Connect, Boomi, and Mule OnDemand.
8. **Security-as-a-service**, is the ability to deliver core security services remotely over the Internet. While typically the security services provided are rudimentary, more sophisticated services are becoming available such as identity management.
   Security-as-a-service providers include Ping Identity.

9. **Management/governance-as-a-service** is any on-demand service that provides the ability to manage one or more cloud services, typically simple things such topology, resource utilization, virtualization, and uptime management. Governance systems are becoming available as well, such the ability to enforce defined policies on data and services.
   Management/governance-as-a-service providers include RightScale, rPath, Xen, and Elastra.
10. **Testing-as-a-service** is the ability to test local or cloud-delivered systems using testing software and services that are remotely hosted. It should be noted that while a cloud service requires testing unto itself, testing-as-a-service systems have the ability to test other cloud applications, Web sites, and internal enterprise systems, and do not require a hardware or software footprint within the enterprise.
    Testing-as-a-service providers include SOASTA.

4. Parallel Computing

Computer software was written conventionally for serial computing. This meant that to solve a problem, an algorithm divides the problem into smaller instructions. These discrete instructions are then executed on the Central Processing Unit of a computer one by one. Only after one instruction is finished, next one starts.

A real-life example of this would be people standing in a queue waiting for a movie ticket and there is only a cashier. The cashier is giving tickets one by one to the persons. The complexity of this situation increases when there are 2 queues and only one cashier.

So, in short, Serial Computing is following:

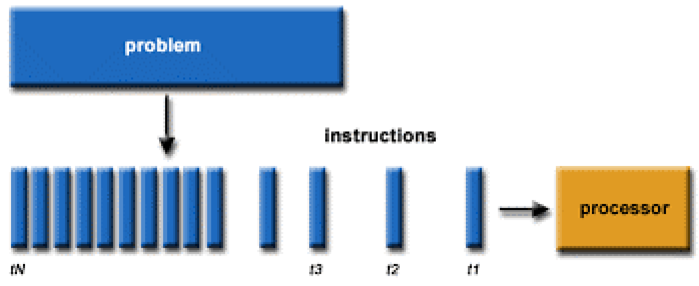
1. In this, a problem statement is broken into discrete instructions.
2. Then the instructions are executed one by one.
3. Only one instruction is executed at any moment of time.

Look at point 3. This was causing a huge problem in the computing industry as only one instruction was getting executed at any moment of time. This was a huge waste of hardware resources as only one part of the hardware will be running for particular instruction and of time. As problem statements were getting heavier and bulkier, so does the amount of time in execution of those statements. Examples of processors are Pentium 3 and Pentium 4.

Now let's come back to our real-life problem. We could definitely say that complexity will decrease when there are 2 queues and 2 cashiers giving tickets to 2 persons simultaneously. This is an example of Parallel Computing.

Parallel Computing :

It is the use of multiple processing elements simultaneously for solving any problem. Problems are broken down into instructions and are solved concurrently as each resource that has been applied to work is working at the same time.



Advantages of Parallel Computing over Serial Computing are as follows:

1. It saves time and money as many resources working together will reduce the time and cut potential costs.
2. It can be impractical to solve larger problems on Serial Computing.
3. It can take advantage of non-local resources when the local resources are finite.
4. Serial Computing 'wastes' the potential computing power, thus Parallel Computing makes better work of the hardware.

Types of Parallelism:

1. **Bit-level parallelism –**
   It is the form of parallel computing which is based on the increasing processor's size. It reduces the number of instructions that the system must execute in order to perform a task on large-sized data.
   Example: Consider a scenario where an 8-bit processor must compute the sum of two 16-bit integers. It must first sum up the 8 lower-order bits, then add the 8 higher-order bits, thus requiring two instructions to perform the operation. A 16-bit processor can perform the operation with just one instruction.
2. **Instruction-level parallelism –**
   A processor can only address less than one instruction for each clock cycle phase. These instructions can be re-ordered and grouped which are later on executed concurrently without affecting the result of the program. This is called instruction-level parallelism.
3. **Task Parallelism –**
   Task parallelism employs the decomposition of a task into subtasks and then allocating each of the subtasks for execution. The processors perform the execution of sub-tasks concurrently.

4. **Data-level parallelism (DLP) –**

Instructions from a single stream operate concurrently on several data – Limited by non-regular data manipulation patterns and by memory bandwidth

Why parallel computing?

- The whole real-world runs in dynamic nature i.e. many things happen at a certain time but at different places concurrently. This data is extensively huge to manage.
- Real-world data needs more dynamic simulation and modeling, and for achieving the same, parallel computing is the key.
- Parallel computing provides concurrency and saves time and money.
- Complex, large datasets, and their management can be organized only and only using parallel computing's approach.
- Ensures the effective utilization of the resources. The hardware is guaranteed to be used effectively whereas in serial computation only some part of the hardware was used and the rest rendered idle.
- Also, it is impractical to implement real-time systems using serial computing.

Applications of Parallel Computing:

- Databases and Data mining.

- Real-time simulation of systems.
- Science and Engineering.
- Advanced graphics, augmented reality, and virtual reality.

Limitations of Parallel Computing:

- It addresses such as communication and synchronization between multiple sub-tasks and processes which is difficult to achieve.
- The algorithms must be managed in such a way that they can be handled in a parallel mechanism.
- The algorithms or programs must have low coupling and high cohesion. But it's difficult to create such programs.
- More technically skilled and expert programmers can code a parallelism-based program well.

Future of Parallel Computing: The computational graph has undergone a great transition from serial computing to parallel computing. Tech giant such as Intel has already taken a step towards parallel computing by employing multicore processors. Parallel computation will revolutionize the way computers work in the future, for the better good. With all the world connecting to each other even more than before, Parallel Computing does a better role in helping us stay that way. With faster networks, distributed systems, and multi-processor computers, it becomes even more necessary.

5. Distributed Computing

A distributed computer system consists of multiple software components that are on multiple computers, but run as a single system. The computers that are in a distributed system can be physically close together and connected by a local network, or they can be geographically distant and connected by a wide area network. A distributed system can consist of any number of possible configurations, such as mainframes, personal computers, workstations, minicomputers, and so on. The goal of distributed computing is to make such a network work as a single computer.

Distributed systems offer many benefits over centralized systems, including the following:

Scalability

The system can easily be expanded by adding more machines as needed.

Redundancy

Several machines can provide the same services, so if one is unavailable, work does not stop. Additionally, because many smaller machines can be used, this redundancy does not need to be prohibitively expensive.

Distributed computing systems can run on hardware that is provided by many vendors, and can use a variety of standards-based software components. Such systems are independent of the underlying software. They can run on various operating systems, and can use various communications protocols. Some hardware might use UNIX or Linux as the operating system, while other hardware might use Windows operating systems. For intermachine communications, this hardware can use SNA or TCP/IP on Ethernet or Token Ring.

Advantages of Distributed Computing.

- Highly efficient
- Scalability
- Less tolerant of failures
- High Availability

Disadvantages of Distributed Systems

Some disadvantages of Distributed Systems are as follows −

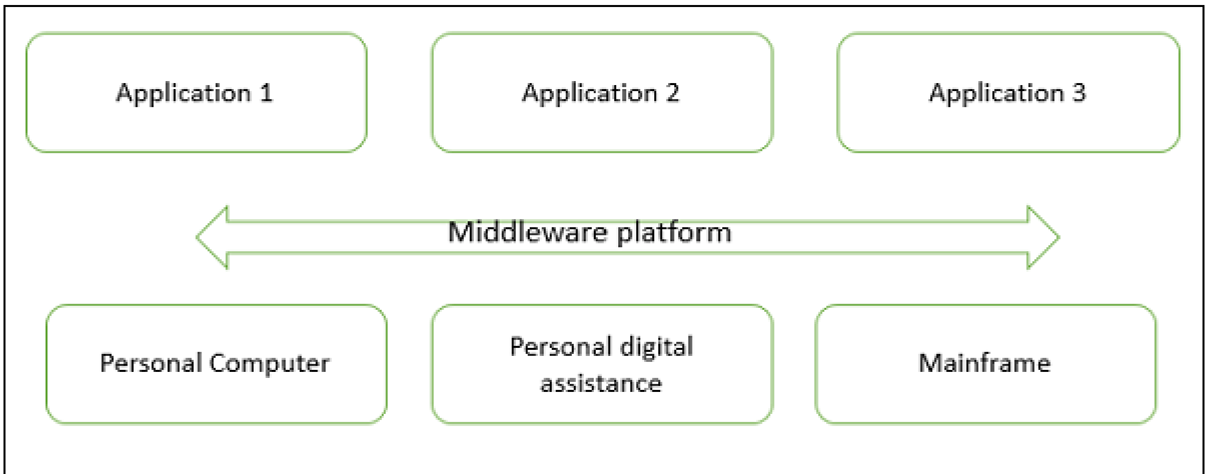
- It is difficult to provide adequate security in distributed systems because the nodes as well as the connections need to be secured.
- Some messages and data can be lost in the network while moving from one node to another.
- The database connected to the distributed systems is quite complicated and difficult to handle as compared to a single user system.
- Overloading may occur in the network if all the nodes of the distributed system try to send data at once.

In distributed architecture, components are presented on different platforms and several components can cooperate with one another over a communication network in order to achieve a specific objective or goal.

- In this architecture, information processing is not confined to a single machine rather it is distributed over several independent computers.

- A distributed system can be demonstrated by the client-server architecture which forms the base for multi-tier architectures; alternatives are the broker architecture such as CORBA, and the Service-Oriented Architecture (SOA).
- There are several technology frameworks to support distributed architectures, including .NET, J2EE, CORBA, .NET Web services, AXIS Java Web services, and Globus Grid services.
- Middleware is an infrastructure that appropriately supports the development and execution of distributed applications. It provides a buffer between the applications and the network.
- It sits in the middle of system and manages or supports the different components of a distributed system. Examples are transaction processing monitors, data convertors and communication controllers etc.

Middleware as an infrastructure for distributed system



Middleware as an infrastructure for distributed system

The basis of a distributed architecture is its transparency, reliability, and availability.

The following table lists the different forms of transparency in a distributed system −

Sr.No.	Transparency & Description
1	**Access** Hides the way in which resources are accessed and the differences in data platform.

2	**Location** Hides where resources are located.
3	**Technology** Hides different technologies such as programming language and OS from user.
4	**Migration / Relocation** Hide resources that may be moved to another location which are in use.
5	**Replication** Hide resources that may be copied at several location.
6	**Concurrency** Hide resources that may be shared with other users.
7	**Failure** Hides failure and recovery of resources from user.
8	**Persistence** Hides whether a resource (software) is in memory or disk.

Advantages

- **Resource sharing** − Sharing of hardware and software resources.
- **Openness** − Flexibility of using hardware and software of different vendors.
- **Concurrency** − Concurrent processing to enhance performance.
- **Scalability** − Increased throughput by adding new resources.
- **Fault tolerance** − The ability to continue in operation after a fault has occurred.

Disadvantages

- **Complexity** − They are more complex than centralized systems.

- **Security** − More susceptible to external attack.
- **Manageability** − More effort required for system management.
- **Unpredictability** − Unpredictable responses depending on the system organization and network load.

6. Unit 4: Advanced Computing Architecture and Framework

Advanced computing is a broad term used to describe either a specific type of high-end computer and the processes undertaken on it, or a set of skills used on personal computers. Both meanings are quite different, and there is no strict definition of the phrase, so that what one person means when they say advanced computing might be very different than what another person says.

Elements of Advanced Computing Systems

Operating advanced computer systems entails working with computer architecture, hardware and software and computer systems. Computer architecture provides a high performance (HPC) potential, while hardware and software is used for high performance graphics. Systems are able to hold a large amount of data and store vast quantities of information. A computer needs the capability to provide networking resources and advanced software.

A computer's infrastructure, combined with an operating system (OS), embeds and distributes information throughout a computer's architecture. This ensures the elements work in conjunction with one another to provide the capability of a computer to operate effectively.

7. Google Cloud for Advanced Cloud Computing

How does the Google Cloud Platform work?

Cloud computing today allows hardware and software products to co-exist remotely (in data centers) and at-scale. Together these products work to deliver specific services. Users typically can access, manage, and use the tools they require via a web-interface – and that's true for Google Cloud Platform services as well.

In addition to service accessibility, users also gain flexibility and choice when working with Google Cloud Platform: Each service is available 'a la carte' so that users can leverage different resources to develop the infrastructure they need.

Once they have identified the Google Cloud Platform services that would benefit them, users simply create a “project” via the intuitive, web-based GCP Console. Better still, project owners can manage which team members or admins have access to which services.

What types of tools are available via the Google Cloud Platform?

Google Cloud Platform services are robust. One way to navigate them is to consider which solutions are available based on your primary computing needs: infrastructure as a service (IaaS), platform as a service (PaaS), and software-as-a-service (SaaS).

• **IaaS** enables IT to run virtual machines without having to invest in or manage this computing infrastructure themselves. Often IT will opt for an IaaS solution when the workload is temporary, experimental, or subject to unexpected changes (e.g. sandbox projects).

• **PaaS** is the next step, building on the IaaS model. Customers opt for all of the benefits of IaaS, plus they get underlying infrastructure – like operating systems and middleware. Their vendor hosts and manages all of these elements.

• **SaaS** goes one more step – everything is available via the web: the provider hosts, manages, and delivers the entire infrastructure including applications. Users simply log in to access the resources the specific solution delivers, e.g. backup and recovery tools.

Another way to navigate Google Cloud Platform is by service-offering type. Core **service categories** include:

1. Compute
2. Networking
3. Storage and Databases
4. Artificial Intelligence (AI) / Machine Learning (ML)
5. Big Data
6. Identity and Security
7. Management Tools

8. Serverless

What is serverless?

Serverless is a cloud computing application development and execution model that enables developers to build and run application code without provisioning or managing servers or backend infrastructure.

Serverless lets developers put all their focus into writing the best front-end application code and business logic they can. All developers need to do is write their application code and deploy it to containers managed by a cloud service provider. The cloud provider handles the rest, provisioning the cloud infrastructure required to run the code and scaling the infrastructure up and down on demand as needed. The cloud provider is also responsible for all routine infrastructure management and maintenance such as operating system updates and patches, security management, capacity planning, system monitoring and more.

Serverless Computing Explained

Serverless is a cloud computing model where the service provider dynamically allocates the exact amount of resources needed on-demand. The main advantage of serverless computing is the provider only charges you for the exact machine resources needed, whereas in cloud computing, you pre-purchase units of bandwidth and resources are dedicated to you at all times, whether they're in use.

While the name serverless is not the elimination of servers from distributed applications. The name serverless refers to the fact that the provider manages, provisions, and maintains backend services on an as-used basis. A company that gets computing resources from a serverless vendor is charged only for the resources their application actually consumes. Hence the client does not have to reserve and pay for a fixed amount of bandwidth or a number of servers, as the service is auto-scaling.

How Serverless Works

Serverless architecture relies on functions, or more specifically functions-as-a-service (FaaS). It is a service model that allows developers to run code directly in the cloud without the need to build packages or maintain any infrastructure. Applications are broken up into individual functions that can be invoked and scaled individually. Listen to the podcast, or read the blog to learn more about serverless stream processing with Apache Kafka®.

Benefits of Serverless

Full scalability – Developers do not have to upload code to servers or do any backend configuration in order to release a working version of an application, nor administrators to upgrade existing servers or add servers. Scalability is automatic and you don’t have to worry about the need to provision the underlying infrastructure.

Easy To Deploy – By using a serverless solution you will get a faster deployment of resources than you would by using other cloud computing models. Instead of weeks and months to deploy an app, you can do it within minutes. The main reason behind such a quick and simplified deployment is that you don’t have to take care of the infrastructure.

No infrastructure requirements – Since developers will be literally using someone else’s computer to execute their serverless functions, there will be no infrastructure to maintain.

No server or software management – Even though ‘serverless’ computing actually takes place on servers, developers are not required to deal with the servers.

Fault-tolerant – Developers are not responsible for ensuring the fault tolerance of the serverless architecture s. The cloud provider provides the IT infrastructure—compute, storage, networking, and databases that will be automatically allocated to account for almost any kind of failure

Lower cost – The costs related to serverless computing are minimal compared to other cloud services, developers are only charged for the duration of execution and not by the server unit. The serverless model enables customers to avoid the costs associated with operating a server such as access authorization, presence detection, security, image processing, and other costs.

No upfront payments – Since you will only be paying for the running code there will be no upfront investment

Reduced Latency – Using serverless functions will bring minimal latency to end-users. As the serverless functions do not operate from a designated origin server, there will not be just one location that an end user's traffic has to be directed to. In this case, all of the cloud provider's data centers will be used and the function will be executed by the nearest server, hence greatly reducing the response time.

Simplified backend code – As the app is not hosted on an origin server, its code can be run from anywhere

Full List of Serverless Providers

The market of serverless architecture providers is no longer limited to major vendors, we have recently seen new players entering this space.

AWS Lambda – an event-driven, serverless computing platform that belongs to Amazon Web Services. It was released in 2014, and now holds the leading position on the market offering the widest range of services available. AWS Lambda can be used for a wide range of tasks and supports code written in Java, Python and Node.js. In addition, the service can launch processes in languages supported by Amazon Linux (includes Bash, Go & Ruby).

AWS Aurora – a relational database engine offered by Amazon Web Services. AWS Aurora is compatible with all the versions of MySQL and PostgreSQ and makes it easy to set up, operate, and scale a relational database in the cloud. It seamlessly grows with business needs.

Azure Functions by Microsoft – the serverless computing service hosted on the Microsoft Azure public cloud The Microsoft Azure service allows you to run small pieces of code in Node.js, C#, Python, PHP and Java and enables running application code on-demand without requiring the provision and management of the underlying infrastructure.

Google Cloud Functions (GCF) – a serverless, event-driven computing service that allows you to create small, single-purpose functions. It became generally available in August 2018 and it caters a series of modular cloud services such as computing, data storage, data analytics and machine learning.

IBM Cloud Functions – a serverless programming platform based on Apache OpenWhisk. IBM Cloud Functions support many programming languages such as JavaScript or Python. It accelerates application development,and allows developers to write out applications on IBM's servers and have them executed remotely and on demand.

Apache OpenWhisk – an open source distributed serverless cloud platform developed by the Apache Software Foundation. It executes functions in response to events that can originate from various sources, like timers, databases, message queues, or websites. It offers per-request scaling and automatic resource allocation, so it can easily adapt to a growing site, service or application.

Spotinst – founded in 2015, Spotinist acquired AWS partner StratCloud. It now offers a complete solution for cloud users by automating multi-cloud infrastructure management. They help enterprises run any workload and support large-scale migrations on any cloud provider.

Cloudflare Workers – offers a suite of products which allows developers to build and extend the capabilities of serverless sites without having to configure or maintain the infrastructure, Cloudflare Workers provides a lightweight JavaScript serverless execution environment that adds performance, security, and reliability for your website.

Oracle Fn Project – an open-source serverless compute platform that you can run anywhere (public, private and hybrid cloud) and supports all programming languages. It has been written in Go and is performant and easy to use.

Kubernetes – an open-source system for container orchestration across multiple hosts. Kubernetes started as a project within Google.in 2014. It is great for deploying all types of apps as it supports all programming languages.

9. Serverless for Application Development

Serverless architectures can be used for many types of applications. For example, you can process transaction orders, analyze click streams, business transactions, evaluate performance data generated by operational IT or manufacturing systems, or telemetry from sensors and other edge devices.

Why build a serverless application?

Building a serverless application allows you to focus on your application code instead of managing and operating infrastructure. You do not have to think about provisioning or configuring servers since AWS handles all of this for you. This reduces your infrastructure management burden and helps you get faster time-to-market.

Building a serverless application offers you four main benefits:

No server management

There is no need to provision or maintain any servers. There is no software or runtime to install, maintain, or administer.

Flexible scaling

Your application can be scaled automatically or by adjusting its capacity through toggling the units of consumption (e.g. throughput, memory) rather than units of individual servers.

High availability

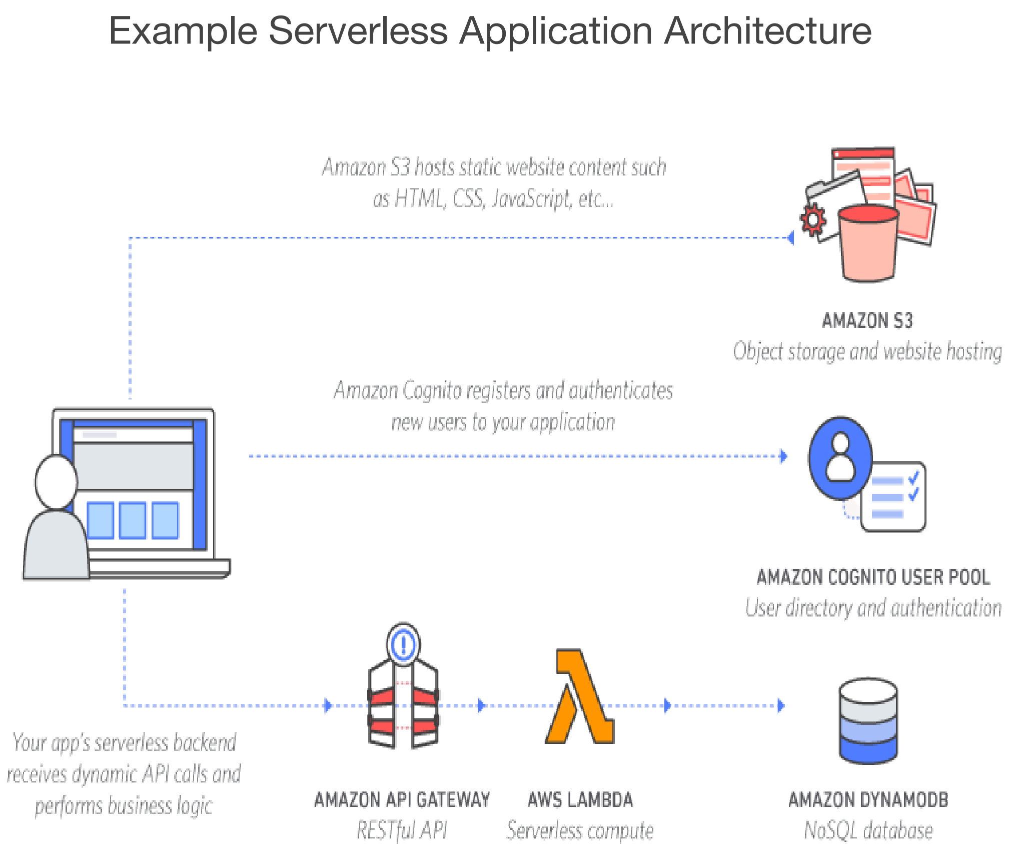
Serverless applications have built-in availability and fault tolerance. You do not need to architect for these capabilities since the services running the application provide them by default.

No idle capacity

You do not have to pay for idle capacity. There is no need to pre- or over-provision capacity for things like compute and storage. For example, there is no charge when your code is not running.

How do I build a serverless application?

You can build a serverless web application by using several AWS services together. Each service is fully managed and does not require you to provision or manage servers. You only need to configure them together and upload your application code to AWS Lambda, a serverless compute service.


Example Serverless Application Architecture
Amazon S3 hosts static website content such as HTML, CSS, JavaScript, etc...
AMAZON S3
Object storage and website hosting
Amazon Cognito registers and authenticates new users to your application
AMAZON COGNITO USER POOL
User directory and authentication
Your app's serverless backend receives dynamic API calls and performs business logic
AMAZON API GATEWAY
RESTful API
AWS LAMBDA
Serverless compute
AMAZON DYNAMODB
NoSQL database

<u>10.Serverless for Analytics</u>

Building a robust data analytics pipeline is a critical step in determining the success (or failure) of your development efforts. Analytics pipelines let you analyze data trends from multiple sources, giving you as complete a picture of your product's success as your data allows. The problem with data analytics work, though, is that it is often not suited to the constant-availability model of traditional software.

This article will serve as a central touchpoint for those looking to build a data pipeline on top of serverless services. We'll define the problem, provide a few examples of the ways in which serverless technology can help, and aggregate useful resources to help you on your next step of the implementation journey.

What Are Data Analytics?

Data Analytics is a practice focused on answering questions about the state of your business. The process of data analytics procures these answers from the application and user behavior data generated by the activity taking place on your applications every day. Data analytics focuses on defining a set of key metrics and tracing those metrics back to a source of truth in the data maintained by your application. These metrics can be as simple as a download count for a specific file, or as complex as a multi-quarter revenue projection model.

Data Analytics in a Serverless Context?

Serverless services, when integrated into your data analytics pipeline, can offer a large array of benefits to your application's metrics reporting. First and foremost is the ability to implement your analytics using an on-demand execution model. Due to the large amount of processing involved, most analytics work lends itself naturally to batching. This model is tailor-made for a serverless architecture, as it allows you to only run the code–and incur charges–when you're actually making use of your computing resources.

Challenges & Considerations?

While serverless technology has a lot to offer data analytics, there are some elements of a robust data pipeline that will be more challenging in an on-demand execution context. The first is the maintainability of the pipeline itself. In a serverless application, each function call happens in isolation. A failed Lambda function has no concept of a call stack unless you build one into it. As such, your developers will need to spend extra time focusing on debuggability, request tracing, and monitoring of your serverless application's behavior to a degree above and beyond that needed for a more traditional web application. Additionally, disaster recovery can in many ways be complicated due to the need to build reporting on third-party communication failures into nearly every API touch point used by your application.

Concepts in Building a Serverless Analytics Pipeline

It is important to note that there is no "perfect" model of a data pipeline. The components, patterns, and behaviors of data analytics pipelines will vary as widely as the applications upon which they are built.

Source of Truth

To begin, it's absolutely vital that you understand the source of truth for every key metric in your system. Where does the data originate? Where does it go? At what point is the data "official," and how is that determination made? Answering these questions for all of your metrics is a time-consuming process, but it is absolutely necessary if you want to draw robust, confident conclusions from your analytics suite.

Data Flow Perspective

When building your application's data model, you'll also want to keep a serverless context in mind. While separation of concerns and reduction of coupling may call for two related objects to be spread across different endpoints, for example, you’re likely to incur higher maintenance costs due to the frequent cross-chatter between the two API locations. Analyze your application from a data flow perspective, and have a feel for the minimal parameter set needed at each data interface. You can use tools like **Amazon Kinesis** to construct robust data flows and data pipelines to power your application, for example, then incorporate **Amazon Athena** to improve your ability to query and analyze data, giving you a more complete view of your application performance.

ETL

Finally, you’ll want to build a robust Extract-Transform-Load (ETL) platform that can not only report on its own health but is able to recover from the inevitable application error. Your ETL pipeline is going to be the linchpin of your analytics suite, as it is responsible for ingesting the data from all applicable sources of truth and converting it into metrics that your business partners can use to plan your organization’s future. A traditional monolith ETL application running on an always-available server will be easy to monitor and institute recovery policies for.

However, upgrading this machine will often represent a prohibitive expense with arcane downtime requirements. While a serverless approach fixes this, the disparate nature of the components of your ETL product introduce additional needed complexity to achieve the same level of monitoring and disaster recoverability as would be present in a comparable traditional application.

Triggers for Data Analytics

The real power of serverless services for data analytics is in triggers. Triggers are actions in a set of online serverless services, such as AWS, that can cause the execution of custom serverless functions. By analyzing the triggers available to your platform and understanding the way that data flows through your application, you can create an efficient pipeline that performs the minimal transformation necessary at each step–and only when the underlying data itself changes!

Caveats and Warnings for Serverless Analytics

While serverless services can provide a massive boost to any data analytics pipeline, they are not without their own pitfalls. Foremost among these is the challenge of monitoring the behavior of your pipeline. As your pipeline grows, the connections between the serverless functions that hold it together grow geometrically, creating an intricate web of behavior that needs to be maintained. If any node of this graph fails, you’ll want to both discover the failure and recover as quickly as possible. This requires extra effort for monitoring and reporting the health of your serverless functions.

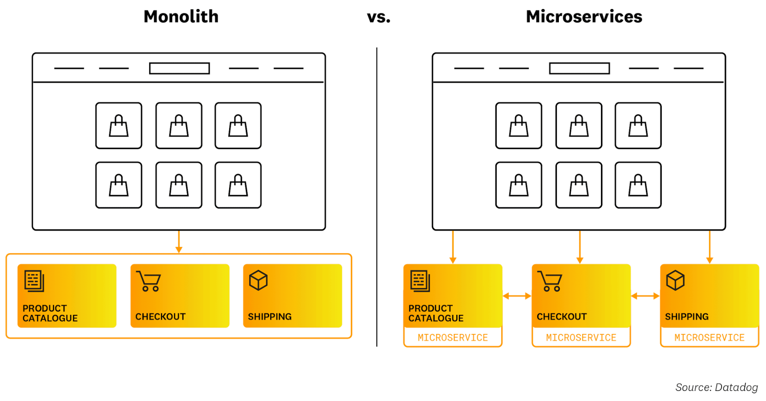
11. Serverless for Microservices Architecture

What Are Serverless Microservices?

Serverless microservices are cloud-based services that use serverless functions to perform highly specific roles within an application. Serverless functions, which execute small segments of code in response to events, are modular and easily scalable, making them well-suited for microservice-based architectures. Serverless functions can also be easily integrated with a range of managed services, which minimizes the overhead that is often associated with other microservice implementations.

Monoliths vs. Microservices

Whereas monolithic applications are built and deployed as one holistic unit, microservice-based applications consist of multiple small services that can be scaled and managed independently of one another. For example, an e-commerce application may include different microservices for the product catalogue, checkout workflow, and shipping process, and each microservice may use its own language, database, and libraries. This design paradigm increases an application's resilience, as an error in one service won't necessarily affect others. Microservices can be hosted on containers, virtual machines, on-premise servers, or serverless functions.



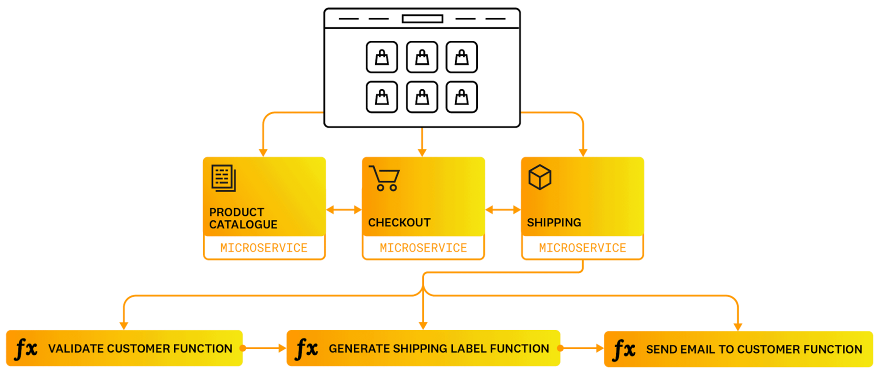
A monolithic application is one holistic unit, whereas microservices break applications down into many independent components.

How Do Serverless Microservices Work?

Serverless microservices are built with serverless functions, which execute small blocks of code in response to HTTP requests, file downloads, database updates, and other events. Cloud providers manage all of the underlying infrastructure that is necessary to run function code, which reduces operational overhead and enables developers to focus on application logic.

A single microservice may include one or more functions that are all deployed at the same time. For instance, a shipping microservice for an ecommerce site might be broken down into a series of functions. When an order is marked as ready to ship, that event could trigger a function that validates the customer's address. A successful validation could trigger another function that generates a shipping label. And finally, creation of the label could trigger a final function that sends a shipping confirmation email to the customer.

Sample Serverless Microservice


PRODUCT CATALOGUE
MICROSERVICE
CHECKOUT
MICROSERVICE
SHIPPING
MICROSERVICE
fx VALIDATE CUSTOMER FUNCTION
fx GENERATE SHIPPING LABEL FUNCTION
fx SEND EMAIL TO CUSTOMER FUNCTION

Source: Datadog

A serverless microservice may include one or more functions that are deployed at the same time.

Benefits and Use Cases of Serverless Microservices

While serverless microservices carry the general advantages of serverless architecture, such as less overhead and improved cost efficiency, their primary benefit is the ease with which you can combine serverless functions with other managed services. Functions can be integrated with databases, message queues, and API management tools in similar ways for different use cases, which means the functions and resources that you use in one microservice can serve as the foundation for other microservices.

Serverless microservices are a good fit for complex, evolving applications, as their modularity makes them easy to manage and scale. Additionally, if your application can be broken down into several discrete services, each of which can in turn be broken down into short-running, event-driven tasks, then it's probably a good candidate for serverless microservices. On the other hand, applications that receive a consistent load or that handle long-running tasks work better as monoliths.

Challenges of Serverless Microservices

Serverless microservices enable engineering teams to iterate quickly and cost-efficiently, but they also pose several challenges:

- **Defining function boundaries**

Teams may find it challenging to define the scope of each function within their microservices.

- **Optimizing function performance**

  Certain performance problems are unique to serverless environments. For example, cold starts, which occur when a function is invoked after a period of inactivity, can lead to increased latency.

- **Monitoring**

  A single application may consist of multiple microservices and short-lived functions that interact with a range of other resources. Given this level of complexity, it can be difficult to trace requests across your environment, understand dependencies, and isolate the root cause of errors. To optimize and troubleshoot serverless microservices, teams need visibility into how their functions and services interact with each other.

12. A. App Engine

What is Google App Engine?

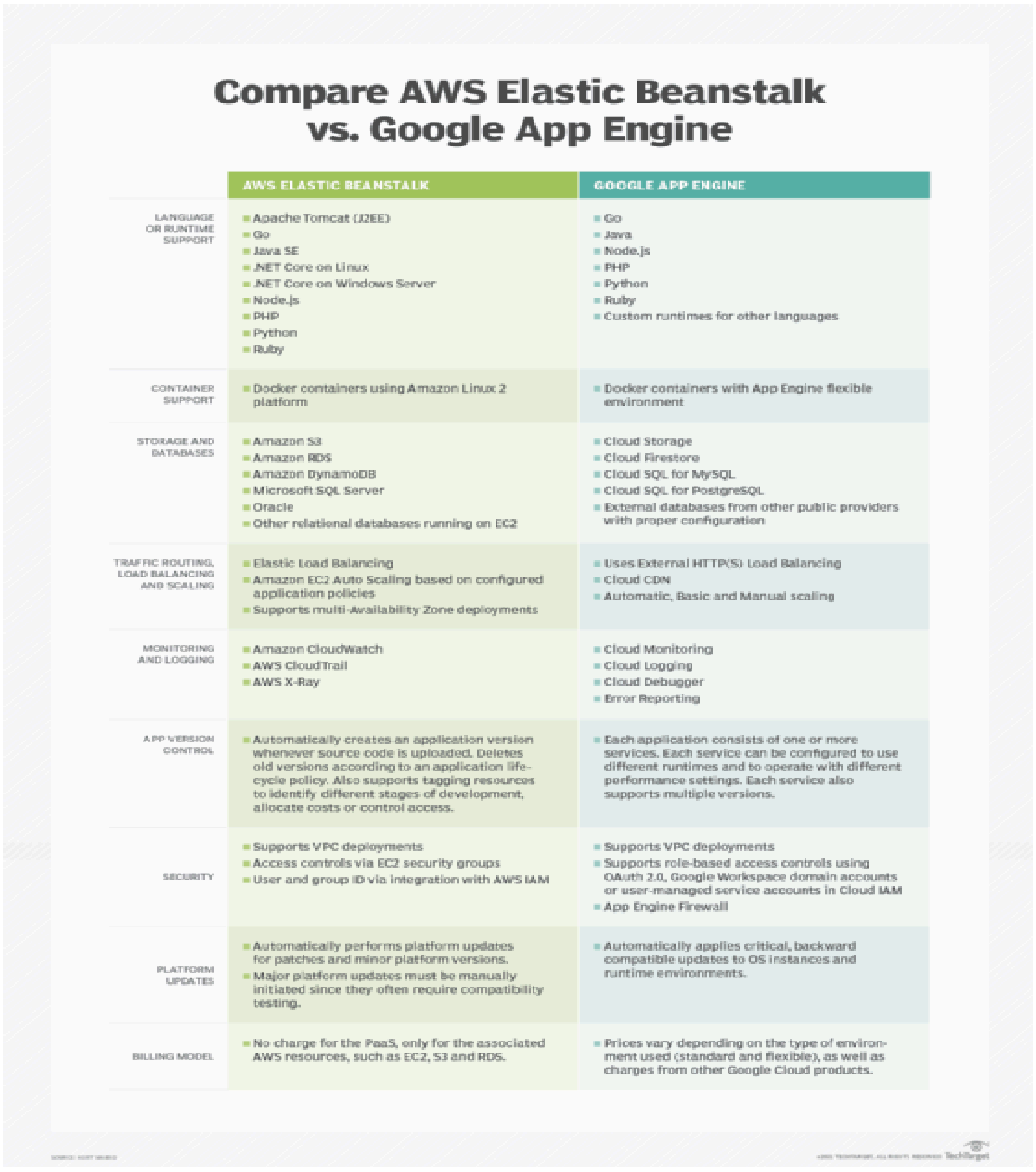
Google App Engine (GAE) is a platform-as-a-service product that provides web app developers and enterprises with access to Google's scalable hosting and tier 1 internet service.

GAE requires that applications be written in Java or Python, store data in Google Bigtable and use the Google query language. Noncompliant applications require modification to use GAE.

GAE provides more infrastructure than other scalable hosting services, such as Amazon Elastic Compute Cloud (EC2). GAE also eliminates some system administration and development tasks to make writing scalable applications easier.
Google provides GAE free up to a certain amount of use for the following resources:

- processor (CPU)
- storage
- application programming interface (API) calls
- concurrent requests

Users exceeding the per-day or per-minute rates can pay for more of these resources.



Compare AWS Elastic Beanstalk vs. Google App Engine

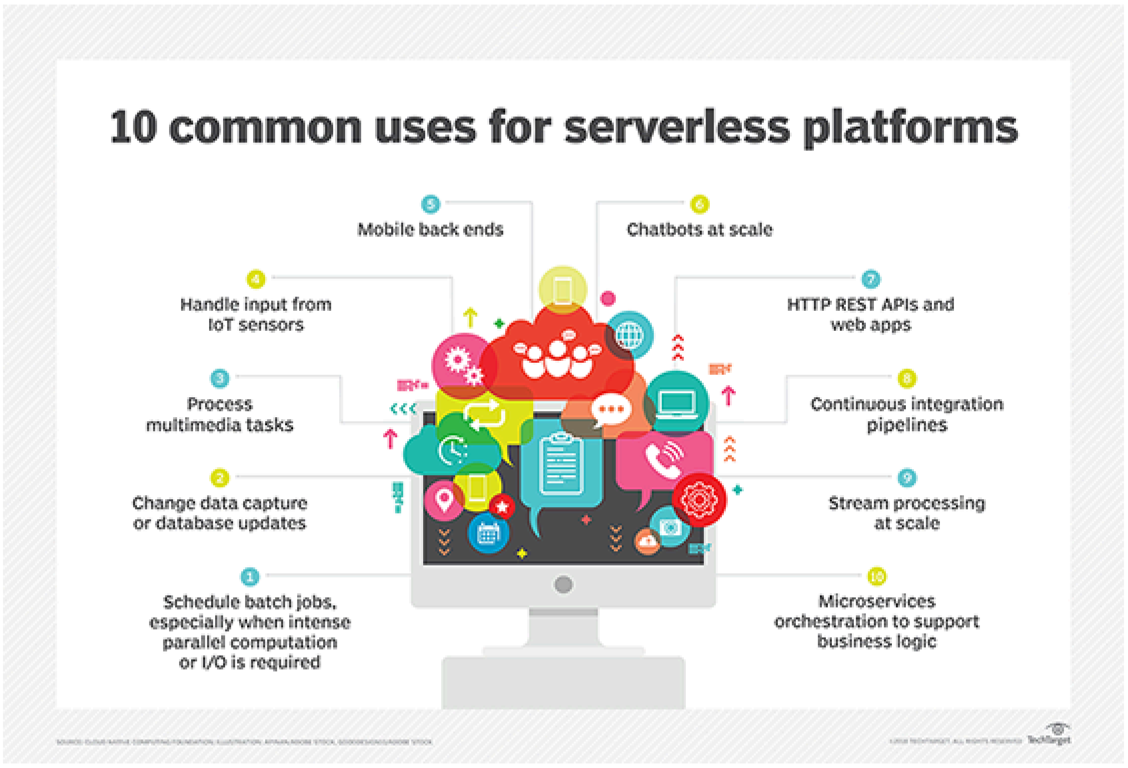
	AWS ELASTIC BEANSTALK	GOOGLE APP ENGINE
LANGUAGE OR RUNTIME SUPPORT	■ Apache Tomcat (J2EE) ■ Go ■ Java SE ■ .NET Core on Linux ■ .NET Core on Windows Server ■ Node.js ■ PHP ■ Python ■ Ruby	■ Go ■ Java ■ Node.js ■ PHP ■ Python ■ Ruby ■ Custom runtimes for other languages
CONTAINER SUPPORT	■ Docker containers using Amazon Linux 2 platform	■ Docker containers with App Engine flexible environment
STORAGE AND DATABASES	■ Amazon S3 ■ Amazon RDS ■ Amazon DynamoDB ■ Microsoft SQL Server ■ Oracle ■ Other relational databases running on EC2	■ Cloud Storage ■ Cloud Firestore ■ Cloud SQL for MySQL ■ Cloud SQL for PostgreSQL ■ External databases from other public providers with proper configuration
TRAFFIC ROUTING, LOAD BALANCING AND SCALING	■ Elastic Load Balancing ■ Amazon EC2 Auto Scaling based on configured application policies ■ Supports multi-Availability Zone deployments	■ Uses External HTTP(S) Load Balancing ■ Cloud CDN ■ Automatic, Basic and Manual scaling
MONITORING AND LOGGING	■ Amazon CloudWatch ■ AWS CloudTrail ■ AWS X-Ray	■ Cloud Monitoring ■ Cloud Logging ■ Cloud Debugger ■ Error Reporting
APP VERSION CONTROL	■ Automatically creates an application version whenever source code is uploaded. Deletes old versions according to an application life-cycle policy. Also supports tagging resources to identify different stages of development, allocate costs or control access.	■ Each application consists of one or more services. Each service can be configured to use different runtimes and to operate with different performance settings. Each service also supports multiple versions.
SECURITY	■ Supports VPC deployments ■ Access controls via EC2 security groups ■ User and group ID via integration with AWS IAM	■ Supports VPC deployments ■ Supports role-based access controls using OAuth 2.0, Google Workspace domain accounts or user-managed service accounts in Cloud IAM ■ App Engine Firewall
PLATFORM UPDATES	■ Automatically performs platform updates for patches and minor platform versions. ■ Major platform updates must be manually initiated since they often require compatibility testing.	■ Automatically applies critical, backward compatible updates to OS instances and runtime environments.
BILLING MODEL	■ No charge for the PaaS, only for the associated AWS resources, such as EC2, S3 and RDS.	■ Prices vary depending on the type of environment used (standard and flexible), as well as charges from other Google Cloud products.

Amazon Elastic Beanstalk is a rival platform as a service to Google App Engine. It is compatible with different programming languages and runtimes than GAE.

How is GAE used?

GAE is a fully managed, serverless platform that is used to host, build and deploy web applications. Users can create a GAE account, set up a software development kit and write application source code. They can then use GAE to test and deploy the code in the cloud.

One way to use GAE is building scalable mobile application back ends that adapt to workloads as needed. Application testing is another way to use GAE. Users can route traffic to different application versions to A/B test them and see which version performs better under various workloads.


10 common uses for serverless platforms
1 Schedule batch jobs, especially when intense parallel computation or I/O is required
2 Change data capture or database updates
3 Process multimedia tasks
4 Handle input from IoT sensors
5 Mobile back ends
6 Chatbots at scale
7 HTTP REST APIs and web apps
8 Continuous integration pipelines
9 Stream processing at scale
10 Microservices orchestration to support business logic

Serverless platforms like Google App Engine are inherently scalable and allow for quick software deployments. See 10 common uses.

What are GAE's key features?

Key features of GAE include the following:

API selection. GAE has several built-in APIs, including the following five:

- **Blobstore** for serving large data objects;
- **GAE Cloud Storage** for storing data objects;
- **Page Speed Service** for automatically speeding up webpage load times;
- **URL Fetch Service** to issue HTTP requests and receive responses for efficiency and scaling; and
- **Memcache** for a fully managed in-memory data store.

Managed infrastructure. Google manages the back-end infrastructure for users. This approach makes GAE a serverless platform and simplifies API management.

Several programming languages. GAE supports a number of languages, including GO, PHP, Java, Python, NodeJS, .NET and Ruby. It also supports custom runtimes.

Support for legacy runtimes. GAE supports legacy runtimes, which are versions of programming languages no longer maintained. Examples include Python 2.7, Java 8 and Go 1.11.

Application diagnostics. GAE lets users record data and run diagnostics on applications to gauge performance.

Security features. GAE enables users to define access policies with the GAE firewall and managed Secure Sockets Layer/Transport Layer Security certificates for free.

Traffic splitting. GAE lets users route requests to different application versions.

Versioning. Applications in Google App Engine function as a set of microservices that refer back to the main source code. Every time code is deployed to a service with the corresponding GAE configuration files, a version of that service is created.

Google App Engine benefits and challenges

GAE extends the benefits of cloud computing to application development, but it also has drawbacks.

Benefits of GAE

- **Ease of setup and use**. GAE is fully managed, so users can write code without considering IT operations and back-end infrastructure. The built-in APIs enable users to build different types of applications. Access to application logs also facilitates debugging and monitoring in production.
- **Pay-per-use pricing**. GAE's billing scheme only charges users daily for the resources they use. Users can monitor their resource usage and bills on a dashboard.
- **Scalability.** Google App Engine automatically scales as workloads fluctuate, adding and removing application instances or application resources as needed.
- **Security**. GAE supports the ability to specify a range of acceptable Internet Protocol (IP) addresses. Users can allowlist specific networks and services and blocklist specific IP addresses.

GAE challenges

- **Lack of control**. Although a managed infrastructure has advantages, if a problem occurs in the back-end infrastructure, the user is dependent on Google to fix it.
- **Performance limits**. CPU-intensive operations are slow and expensive to perform using GAE. This is because one physical server may be serving several separate, unrelated app engine users at once who need to share the CPU.
- **Limited access.** Developers have limited, read-only access to the GAE filesystem.
- **Java limits.** Java apps cannot create new threads and can only use a subset of the Java runtime environment standard edition classes.

12.B. Google Big Query

Google BigQuery is a serverless, highly scalable data warehouse that comes with a built-in query engine. The query engine is capable of running SQL queries on terabytes of data in a matter of seconds, and petabytes in only minutes. You get this performance without having to manage any infrastructure and without having to create or rebuild indexes.

BigQuery has legions of fans. Paul Lamere, a Spotify engineer, was thrilled that he could finally talk about how his team uses BigQuery to quickly analyze large datasets: "Google's BigQuery is *da bomb*," he tweeted in February 2016. "I can start with 2.2Billion 'things' and compute/summarize down to 20K in < 1 min." The scale and speed are just two notable features of BigQuery.
For example, Twitter recently reported in its blog that it was able to democratize data analysis with BigQuery by providing some of its most frequently used tables to Twitter employees from a variety of teams (Engineering, Finance, and Marketing were mentioned).

Imagine that you run a chain of equipment rental stores. You charge customers based on the length of the rental, so your records include the following details that will allow you to properly invoice the customer:

1. Where the item was rented

2. When it was rented
3. Where the item was returned
4. When it was returned

Perhaps you record the transaction in a database every time a customer returns an item.1

From this dataset, you would like to find out how many “one-way” rentals occurred every month in the past 10 years. Perhaps you are thinking of imposing a surcharge for returning the item at a different store and you would like to find out what fraction of rentals would be affected. Let’s posit that wanting to know the answer to such questions is a frequent occurrence—it is important for you to be able to answer such ad hoc questions because you tend to make data-driven decisions.

Some of the key features of Google BigQuery are as follows:

- Scalable Architecture
- Faster Processing
- Fully-Managed
- Security
- Real-time Data Ingestion
- Fault Tolerance
- Pricing Models

1) Scalable Architecture

BigQuery has a scalable architecture and offers a petabyte scalable system that users can scale up and down as per load.

2) Faster Processing

Being a scalable architecture, BigQuery executes petabytes of data within the stipulated time and is more rapid than many conventional systems. BigQuery allows users to run analysis over millions of rows without worrying about scalability.

3) Fully-Managed

BigQuery is a product of the Google Cloud Platform, and thus it offers fully managed and serverless systems.

4) Security

BigQuery has the utmost security level that protects the data at rest and in flight.

5) Real-time Data Ingestion

BigQuery can perform real-time data analysis, thereby making it famous across all the IoT and Transaction platforms.

6) Fault Tolerance

BigQuery offers replication that replicates data across multiple zones or regions. It ensures consistent data availability when the region/zones go down.

7) Pricing Models

The Google BigQuery platform is available in both **on-demand** and **flat-rate subscription models**. Although **data storage** and **querying** will be **charged**, **exporting, loading,** and **copying data** is **free**. It has separated computational resources from storage resources. You are only charged when you run queries. The quantity of data processed during searches is billed.

You can use BigQuery for the following use cases:

- Interacting with BigQuery
- Running and Managing Jobs
- Working with datasets
- Working with table schemas
- Working with tables
- Working with partitioned tables
- Working with clustered tables
- Working with table snapshots

- Working with views
- Working with materialized views and many more.

12.C. Cloud Datastore

Google **Cloud Datastore** is a NoSQL database with high scalability for the applications. The purpose of Cloud Datastore is to handle replication and sharding aspects, to give you a durable and available database for automatic scaling of the load embedded over applications. Google Cloud Datastore offers high-end capabilities that include SQL-like queries, ACID transactions, indexes, and others, to help enhance the end outcomes. It is not just feasible concerning your scaling aspects but also takes care of application development and high performance.

Cloud Datastore is rich with high-end functionalities to carry out atomic transactions. The datastore has the potential to execute several sets of operations. Under these transactions, you can ensure to get either all of the operations to work seamlessly or all to fail! There is nothing in between! The storage and data querying potential is commendable with Google Cloud Datastore. It maps efficiently to Object-Oriented Languages and Scripting languages, with natural measures. And it is then exposed to the applications through diverse clients.

Working Overview of Cloud Datastore

Cloud Datastore is meant for applications that demand reliability upon the highly available structured data at a fixed scale. You can make use of the Google Cloud Datastore to store & query different types of data that include product catalogs, user profiles, and transactions. The product catalogs that intend to offer real-time inventory and other associated product details for the retailer can be stored and queried within Cloud Datastore.

The user profile data is often intended to deliver customized experiences based upon past preferences and activities. This data is then stored and queried with Cloud Datastore for the required database. Datastore is not suitable for all of the use cases, such as analytic data. Datastore doesn't have the properties of a relational database, which makes it inefficient for the analytics data!

At high-end levels, Cloud Datastore is best suitable for storing the transactions and hierarchical data. All of this stored data consists of a flexible and non-relational schema. There is a specific range of requirements for you to use Datastore over your web application. Therefore, if the web application requirement matches any of the below cases, then you can go ahead with the adaptation of Cloud Datastore:

- If you are in need of any amount of less or more storage capacity, Datastore can help you with it! Datastore has the potential to handle data storage from Kilobytes to Petabytes. And there will be no fluctuation in the performance for it!
- ACID (Atomicity, Consistency, Isolation, and Durability) compliant and multi-document transactions are supported with Cloud Datastore.
- Cloud Datastore has the potential to help support the primary, secondary & composite indexes. If you are willing to know more about indexes, you can refer to this specific Google documentation upon Indexes, for the same!
- Datastore by Google Cloud encrypts all of the data automatically before it can be written over the disk. And it is also offering Identity & Access Management (IAM). It is important for implementation upon the web applications to permit accessibility to individuals for accessing specific resources or to prevent someone from accessing select resources. To get a better idea of Identity and Access Management protocol within Google Cloud and Datastore, refer to this official documentation for IAM.

Google **Cloud Datastore** offers two redundancy levels that depend upon the replications within multiple locations. The levels include Regional replication and multi-region replication:

Under regional replication, the data undergoes replication within at least 3 varying zones but within that same region. Hence, this will make that database more resilient towards zonal outages. Regional replication is preferable for implementing low write latency. Under it, you can prefer co-locating the computing machines of the application within that same region.

The multi-region replication allows replication of data, within multiple zones, across a minimum of two regions. Hence, this brings out the result of enhanced availability and redundancy. But in the resulting outcome, the write latency gets higher. A witness node is proposed and deployed within the third region to be the tiebreaker between every two regions.

Hence, the abilities of Google **Cloud Datastore** databases are efficient and are productive for the users to pick. Datastore has the ability to meet diverse requirements for web applications, which makes it an ideal choice for several use cases that include:

1. User Profiles

2. Real-Time Inventories

3. State Mutations

4. User Session Management

Comparison Between the Datastore, Firestore, and Relational Database

Concept	**Datastore**	**Firestore**	**Relational database**
Category of object	Kind	Collection group	Table
One object	Entity	Document	Row
Individual data for an object	Property	Field	Column
Unique ID for an object	Key	Document ID	Primary Key

To be more precise, here are the ways with which you can consider Datastore to be different from that of the traditional or relational database:

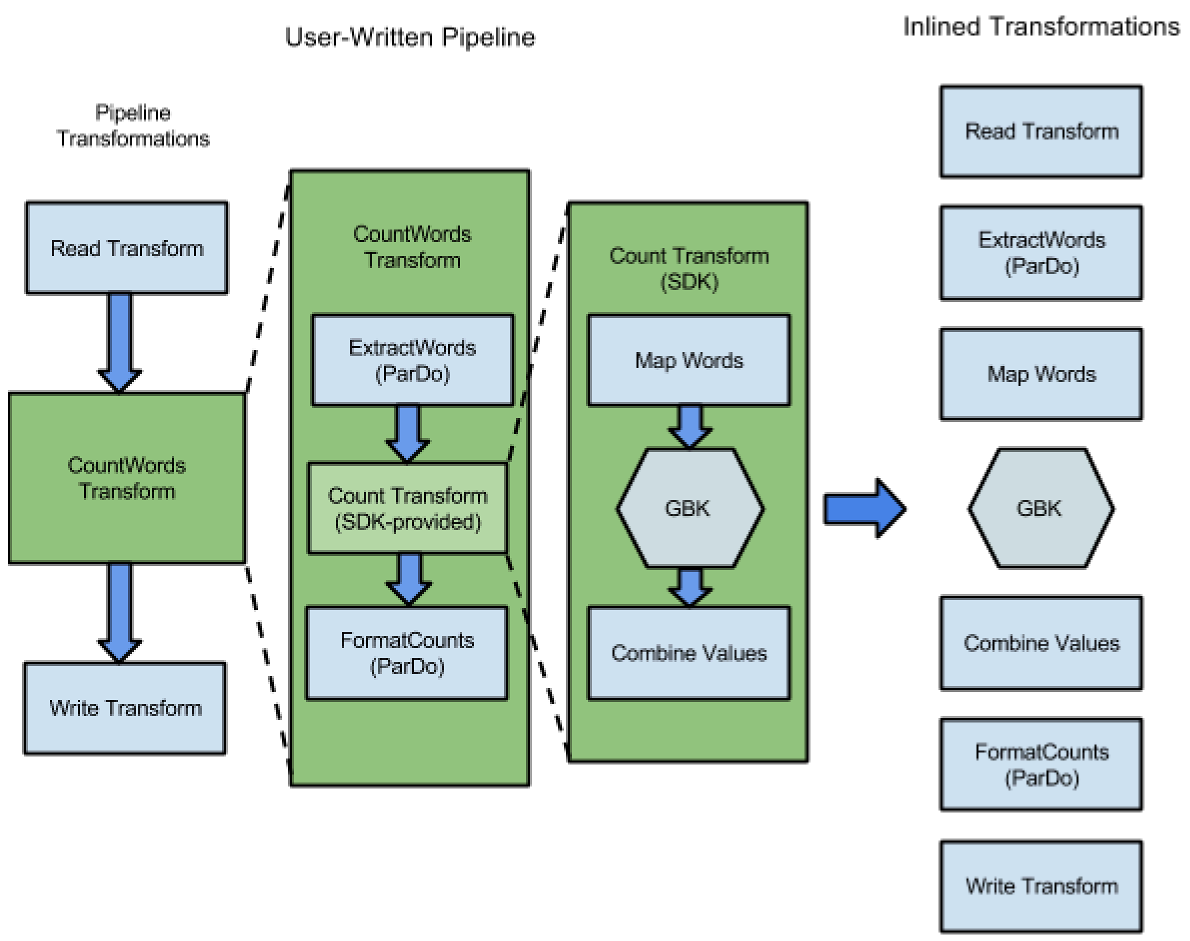
- Cloud Datastore has the potential to implement auto-scaling for the larger data sets. Hence, it allows the applications to emit high performance whenever they receive high traffic.
- Cloud Datastore intends to write scale by distributing the data automatically, whenever necessary.

- Cloud Datastore doesn't intend to offer support for inequality filtering, multiple properties, or to join operations. Apart from that, Cloud Datastore is not even offering filtering aspects based upon data over the subquery results.
- Cloud Datastore is completely schemaless, which is not the case with a traditional relational database. There is no necessity for entities that resembles the same kind to have consistent property sets.

12.D. Cloud Dataflow

Google Cloud Dataflow is a managed service used to execute **Apache Beam data processing pipelines** using the Google Cloud Platform (GCP) ecosystem.

For our purposes, **data processing** refers to the procedure of taking large amounts of data, potentially combining it with other data, and ending with an enriched dataset of similar size or a smaller summary dataset. A **pipeline** is a sequence of steps that reads, transforms, and writes data.



A

Dataflow pipeline from Beam's WordCount Example.

Apache Beam is a framework with bindings in Python and Java that enables simple representation of data processing pipelines. Beam is built around the concept of **PCollections** (parallel collections) and **Transforms** on those collections. Together, PCollections and Transforms form Apache Beam Pipelines.

Beam is based on an internal model at Google, which evolved from MapReduce and successors like Flume & MillWheel. The Beam model represents all datasets uniformly via PCollections. A PCollection could be in-memory, read from Cloud Storage, queried from BigQuery, or read as a stream from a **Pub/Sub** topic.

Dataflow primarily functions as a fully managed pipeline runner for Apache Beam. Dataflow was released in April 2015. In January 2016, Google donated the underlying SDK and a few other components to the Apache Software Foundation. The donated code formed the basis for the Apache Beam project, hence the close connection to Dataflow.

The Beam documentation provides conceptual information and reference material for its programming model, SKDs, and other runners.

Overview & Functionality of Google Cloud Dataflow

Now that we understand the relationship between Beam and Dataflow, we can discuss Dataflow functionality. Dataflow is an expansive system designed to make **data and analytics** more accessible using **parallel processing**. It has a board range of use-cases including:

- **Data integration and preparation** (e.g. preparing data for interactive SQL in BigQuery),
- Examining a **real-time stream** of events for significant patterns,
- Implementing advanced processing pipelines to **extract insights**.

Unlike other pipeline runners, Cloud Dataflow requires no initial setup of underlying resources: it’s a fully managed runner. Because Dataflow is fully integrated with the **Google Cloud Platform** (GCP), it can easily combine services we’ve discussed in other articles, like **Google BigQuery**.

Dataflow Concepts

PCollections

A ***PCollection*** is the canonical representation of 'data' in a pipeline. Every step in a pipeline takes and/or returns a PCollection object. PCollections can hold any type of object—**integers,**

strings, composite objects, table rows, etc. without any size restrictions. There are two types of PCollections

- ***Bounded* PCollections** hold datasets of limited and known size that do not change. For example, data sources and sinks for the classes TextIO, BigQueryIO, and DatastoreIO, and those created using the custom source/sink API are Bounded PCollections.
- ***Unbounded* PCollections** are representations of data that are continuously added, i.e. there are no boundaries to the dataset. PubSubIO works with unbounded PCollections on the source and sink sides. BigQueryIO accepts unbounded PCollections as sink information.

Since Unbounded PCollections can contain an *unlimited* amount of data, interruption points must be defined to allow Dataflow to work on finite chunks of transformed information.

Transforms

Transforms represent each step taken to mutate data from the original source to a modified state. Transforms primarily occur on PCollections. *PTransform* objects implement the *apply* method, which is where transformations are applied in Apache Beam.

Dataform best practices dictate breaking down Transforms into chunks of logic that can be grouped together. This improves code readability, testability, and reusability. Dataform allows for viewing pipelines as a Directed Acyclic Graph (DAG) and building discrete Transforms aids in a digestible visual representation of a data pipeline.

ParDo

ParDo is one of the core transformations for **parallel computing** in Dataform. It applies logic specified via a **function class** (DoFn) to each element in a PCollection. ParDo can accept multiple input data through side inputs & return multiple output PCollections.

GroupByKey

GroupByKey allows for grouping key-value pair collections in parallel by their key. GroupByKey is often useful in one-to-many datasets that should be grouped by key—it returns a collection of pairs with unique keys holding multiple values.

Combine

Combine aggregates a *collection* of values into a *single* value or key-value pairs into key 'grouped' collections. The transformation also accepts a function class that can be used to aggregate a collection.

Flatten

Flatten is a common transformation for **merging a list of PCollection objects** (of the same type) into a single PCollection.

Data Sources & Sinks for Google Cloud Dataflow

Data pipelines read from a data source and write to a sink. Cloud Dataflow makes it simple to treat many sources/sinks similarly by providing a set of representative interfaces, allowing for flexible information processing.

Sources generate PCollections and sinks accept them as input during a write operation. Here are some common Dataflow sources and sinks:

- **Cloud Storage Datasets**: Cloud Dataflow can accept and write to Google Cloud Storage (GCS) datasets. The tight integration with other GCP resources is one of Dataflow's biggest strengths.
- **BigQuery Tables**: The BigQueryIO class allows for interaction with Google BigQuery for reading and writing data. BigQuery can be a useful sink if further aggregation or analysis is required on data.
- **Google Cloud Pub/Sub Messages**: While only available for streaming pipelines, Dataflow can read from and write data to Cloud Pub/Sub messages with the PubSubIO class. Pub/Sub is very powerful for real-time data consumption.

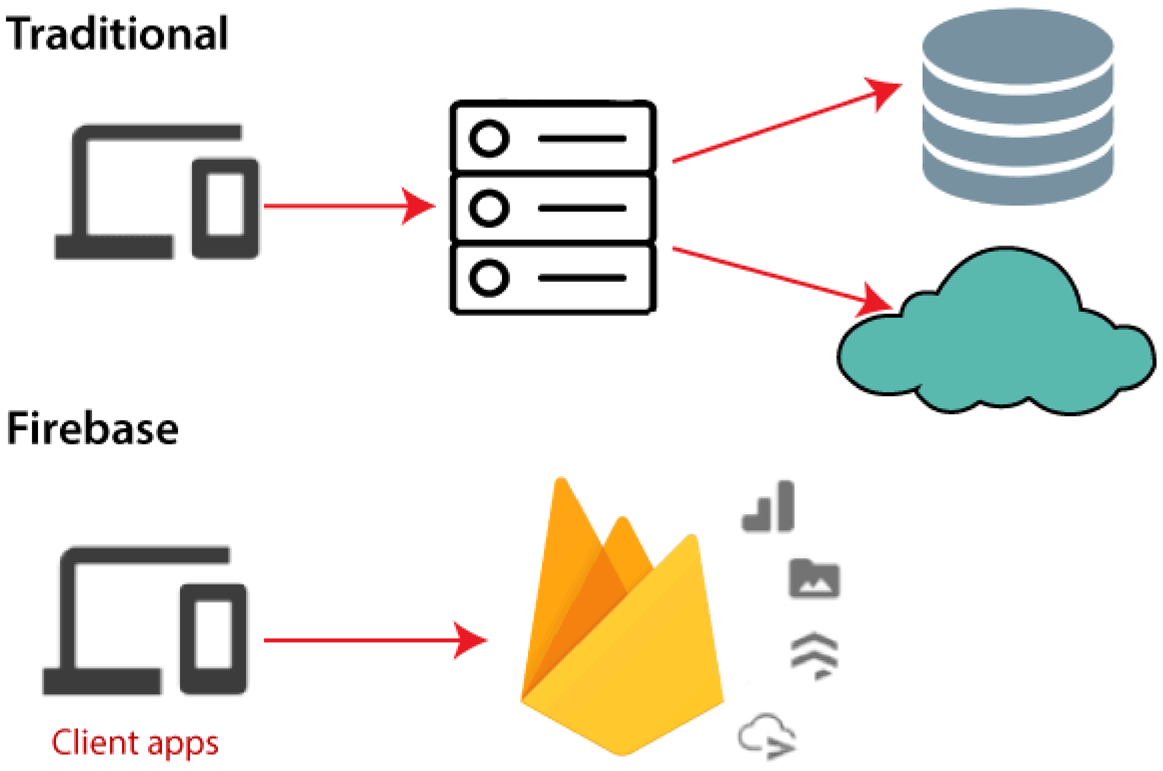
13.A. Firebase

Firebase tutorial is designed for both beginners and professionals. Our tutorial provides all the basic and advanced services knowledge, such as Real-time Database, Cloud Messaging, Hosting and Crash Reporting, etc.

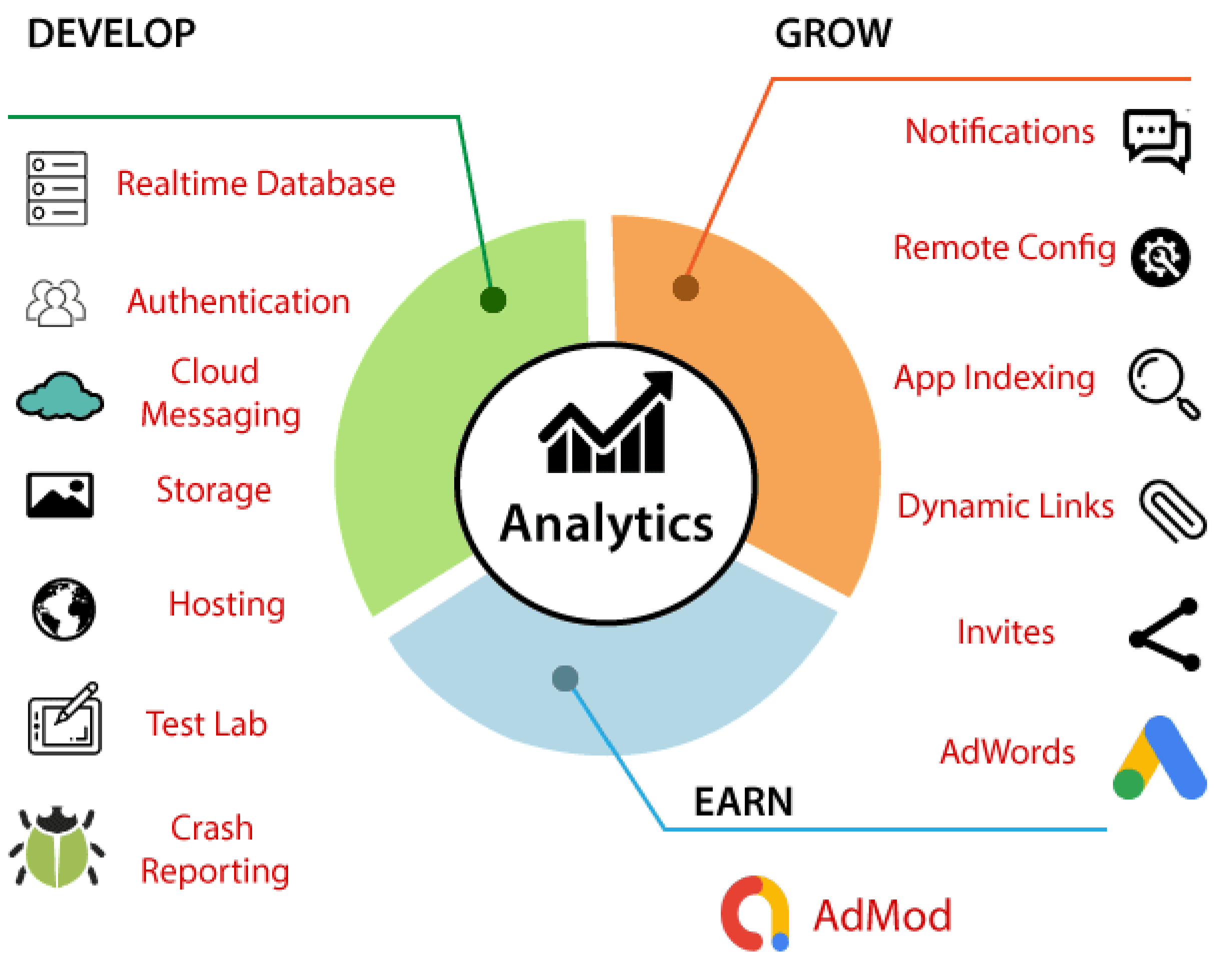
Firebase is a Backend-as-a-Service, and it is a real-time database which is basically designed for mobile applications. This tutorial is designed in such a way that we can easily understand or can perform the service of Firebase in a very efficient way.

Introduction

Firebase is a Backend-as-a-Service(BaaS) which started as a YC11 startup. It grew up into a next-generation app-development platform on Google Cloud Platform. Firebase (a NoSQLjSON database) is a real-time database that allows storing a list of objects in the form of a tree. We can synchronize data between different devices.



Google Firebase is Google-backed application development software which allows developers to develop **Android, IOS,** and **Web apps**. For reporting and fixing app crashes, tracking analytics, creating marketing and product experiments, firebase provides several tools.

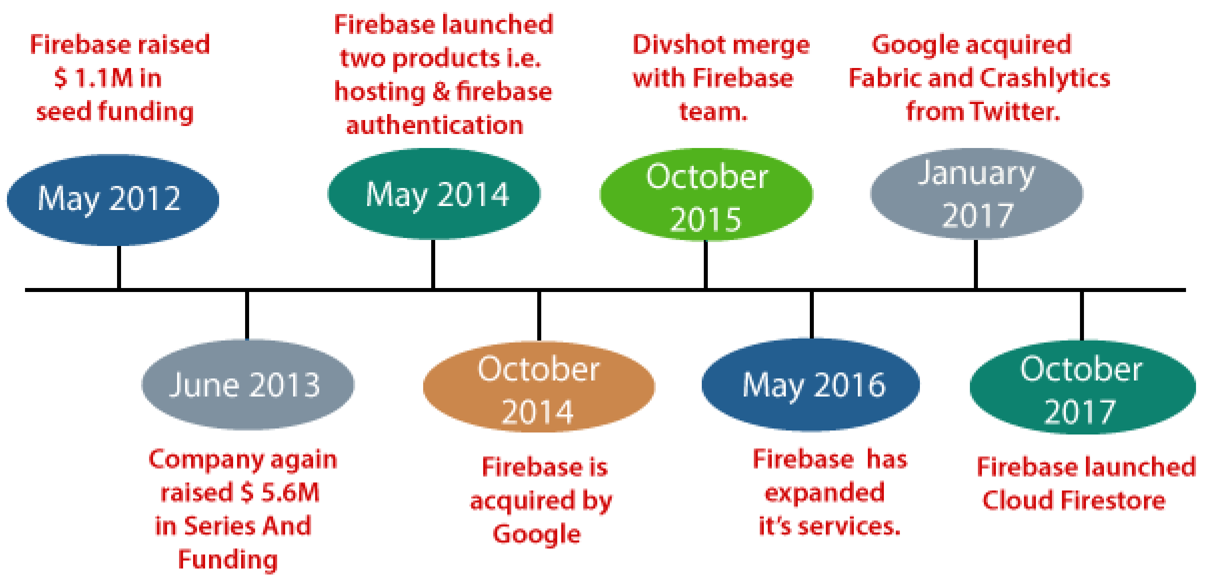


Firebase has three main services, i.e., a real-time database, user authentication, and hosting. We can use these services with the help of the Firebase iOS SDK to create apps without writing any server code.

History of Firebase

Firebase evolved from **Envolve**. Envolve is a prior startup founded by **James Tamplin** and **Andrew Lee** in 2011. Envolve provided developers an API which allowed the integration of online chat functionality into their websites. After releasing the chat service, it found that the envlove was being used to pass application data, which were not chat messages. Developers used Envolve to sync application to separate the real-time architecture and the chat system which powered it. In September 2011, Tamplin and Lee founded firebase as a separate company. It was lastly launched to the public in April 2012.

Firebase Real-time Database was the first product of firebase. It is an API which syncs application data across Android, iOS, and Web devices. It gets stored on Firebase's cloud. Then the firebase real-time database helps the developers to build real-time, collaborative applications.


Firebase raised $ 1.1M in seed funding
May 2012
June 2013
Company again raised $ 5.6M in Series And Funding
Firebase launched two products i.e. hosting & firebase authentication
May 2014
October 2014
Firebase is acquired by Google
Divshot merge with Firebase team.
October 2015
May 2016
Firebase has expanded it's services.
Google acquired Fabric and Crashlytics from Twitter.
January 2017
October 2017
Firebase launched Cloud Firestore

- In May 2012, after launching the beta, Firebase raised $1.1M in seed funding from Greylock Partners, venture capitalists Flybridge Capital Partners, New Enterprise Associates, and Founder Collective.
- In June 2013, the company again raised $5.6M in **Series A funding** from Flybridge Capital Partnersandventure capitalists Union Square Ventures.
- Firebase launched two products in 2014, i.e., Firebase Hosting and Firebase Authentication. It positioned the company as a mobile backend as a service.
- Firebase was acquired by Google in October 2014.
- Google promoted Divshot to merge it with the Firebase team in October 2015.
- In May 2016, Firebase expanded its services to become a unified platform for mobile developers. Now it has integrated with various other Google services, including AdMob, Google Cloud Platform, and Google Ads, to offer broader products and scale it for developers.
- Google acquired Fabric and Crashlytics from Twitter in January 2017 to add Fabric and Crashlytics services to Firebase.
- Firebase launched Cloud Firestore in October 2017. It is a realtime document database as the successor product for the original Firebase Realtime Database.

Why use Firebase?

- Firebase manages real-time data in the database. So, it easily and quickly exchanges the data to and from the database. Hence, for developing mobile apps such as live streaming, chat messaging, etc., we can use Firebase.
- Firebase allows syncing real-time data across all devices - iOS, Android, and Web - without refreshing the screen.
- Firebase provides integration to Google Advertising, AdMob, Data Studio, BigQuery DoubleClick, Play Store, and Slack to develop our apps with efficient and accurate management and maintenance.

- Everything from databases, analytics to crash reports are included in Firebase. So, the app development team can stay focused on improving the user experience.
- Firebase applications can be deployed over a secured connection to the firebase server.
- Firebase offers a simple control dashboard.
- It offers a number of useful services to choose from.

Pros and Cons of Firebase

Firebase has a lot of pros or advantages. Apart from the advantages, it has disadvantages too. Let's take a look at these advantages and disadvantages:

Pros

- Firebase is a real-time database.
- It has massive storage size potential.
- Firebase is serverless.
- It is highly secure.
- It is the most advanced hosted BaaS solution.
- It has minimal setup.
- It provides three-way data binding via angular fire.
- It provides simple serialization of app state.
- We can easily access data, files, auth, and more.
- There is no server infrastructure required to power apps with data.
- It has JSON storage, which means no barrier between data and objects.

Cons

- Firebase is not widely used, or battle-tested for enterprises.
- It has very limited querying and indexing.
- It provides no aggregation.
- It has no map-reduce functionality.
- It cannot query or list users or stored files.

13.B. Cloud Pub/ Sub

Pub-Sub is a real-time messaging service by google cloud for applications to publish and subscribe to the events.
One can publish the message manually or watch the changes in the Gmail using Gmail API and notify the user application on a realtime basis.

The core components of Pub-Sub include a **publisher**, **subscription**, and a **topic**.

- **Topic**: A resource to which messages are sent by publishers.
- **Publisher:** A publisher creates a topic in the Pub-Sub service and sends messages to the topic.
- **Subscription**: A resource containing messages from a topic, which would be delivered to the subscribing user application.

Types of Pub/Sub services

Pub/Sub consists of two services:

- **Pub/Sub service.** This messaging service is the default choice for most users and applications. It offers the highest reliability and largest set of integrations, along with automatic capacity management. Pub/Sub guarantees synchronous replication of all data to at least two zones and best-effort replication to a third additional zone.
- **Pub/Sub Lite service.** A separate but similar messaging service built for lower cost. It offers lower reliability compared to Pub/Sub. It offers either zonal or regional topic storage. Zonal Lite topics are stored in only one zone. Regional Lite topics replicate data to a second zone asynchronously. Also, Pub/Sub Lite requires you to pre-provision and manage storage and throughput capacity. Consider Pub/Sub Lite only for applications where achieving a low cost justifies some additional operational work and lower reliability

13.C. Cloud Functions

What are Google Cloud Functions?

Google Cloud Functions is a serverless execution environment for building and connecting cloud services. With Cloud Functions you write simple, single-purpose functions that are attached to events emitted from your cloud infrastructure and services. Your function is triggered when an event being watched is fired. Your code executes in a fully managed environment. There is no need to provision any infrastructure or worry about managing any servers.

You can write Cloud Functions using a number of supported programming languages. You can take your function and run it in any standard runtime environment for one of the supported languages, which makes both portability and local testing a breeze.

Cloud Functions offers two product versions: Cloud Functions (1st gen), the original version, and Cloud Functions (2nd gen), a new version built on Cloud Run and Eventarc to provide an enhanced feature set. See Cloud Functions version comparison for more information.

Connect and extend cloud services

Cloud Functions provides a connective layer of logic that lets you write code to connect and extend cloud services. Listen and respond to a file upload to Cloud Storage, a log change, or an incoming message on a Pub/Sub topic. Cloud Functions augments existing cloud services and allows you to address an increasing number of use cases with arbitrary programming logic. Cloud Functions have access to the Google Service Account credential and are thus seamlessly authenticated with the majority of Google Cloud services, including Cloud Vision, as well as many others. In addition, Cloud Functions are supported by numerous Google Cloud client libraries, which further simplify these integrations.

Events and triggers

Cloud *events* are things that happen in your cloud environment. These might be things like changes to data in a database, files added to a storage system, or a new virtual machine instance being created.

Events occur whether or not you choose to respond to them. You create a response to an event with a *trigger*. A trigger is a declaration that you are interested in a certain event or set of events. Binding a function to a trigger allows you to capture and act on events. For more information on creating triggers and associating them with your functions, see Cloud Functions triggers.

Serverless

Cloud Functions removes the work of managing servers, configuring software, updating frameworks, and patching operating systems. The software and infrastructure are fully managed by Google so that you just add code. Furthermore, provisioning of resources happens automatically in response to events. This means that a function can scale from a few invocations a day to many millions of invocations without any work from you.

Use cases

Asynchronous workloads (such as lightweight ETL) or cloud automations (such as triggering application builds) now no longer need their own server or a developer to manually manage them. You simply deploy a function bound to the event you want and you're done.

The fine-grained, on-demand nature of Cloud Functions also makes it a perfect candidate for lightweight APIs and webhooks. In addition, the automatic provisioning of HTTP endpoints when you deploy an HTTP function means there is no complicated configuration required as there is with some other services. See the following table for additional common Cloud Functions use cases:

Use case	**Description**
Data processing / ETL	Listen and respond to Cloud Storage events such as when a file is created, changed, or removed. Process images, perform video transcoding, validate and transform data, and invoke any service on the internet from your Cloud Functions.
Webhooks	Via a simple HTTP trigger, respond to events originating from 3rd party systems like GitHub, Slack, Stripe, or from anywhere that can send HTTP requests.
Lightweight APIs	Compose applications from lightweight, loosely coupled bits of logic that are quick to build and that scale instantly. Your functions can be event-driven or invoked directly over HTTP/S.
Mobile backend	Use Google's mobile platform for app developers, Firebase, and write your mobile backend in Cloud Functions. Listen and respond to events from Firebase Analytics, Realtime Database, Authentication, and Storage.
IoT	Imagine tens or hundreds of thousands of devices streaming data into Pub/Sub, thereby launching Cloud Functions to process, transform and

Use case	Description
	store data. Cloud Functions lets you do it in a way that's completely serverless.

14.Unit 5: Role of Cloud Computing in An AI Implementation

Cloud computing environment and solutions are enabling enterprises to become more agile, flexible, and cost-effective as this substantially reduces infrastructure management costs for enterprises, Dipanshu and Swathi state. Artificial Intelligence (AI) enables additional flexibility as it helps them manage large data repositories, streamline data, optimize workflows, and produce real-time insights to transform day-to-day operations and re-imagine end customer experience.

The Unification of AI and Cloud Computing

The application of AI software based on machine learning (ML) algorithms in cloud environments delivers intuitive and connected experiences for customers and users. Alexa and Siri are but two examples of this seamless combination that enables a variety of operations from conducting a search to playing a song to making a purchase.

n ML models, large sets of data are used to train the algorithm. This data could be structured, unstructured, or raw and needs powerful CPUs and GPUs to process. Only an ideal combination of public, private, or hybrid cloud systems (based on security and compliance requirements) can provide such huge amounts of compute power today. Further, the cloud also enables services that are used in ML, such as serverless computing, batch processing, and container orchestration.

With public cloud services, developers don’t need to build and manage a separate infrastructure for hosting AI platforms. They can use ready configurations and models to test and deploy AI applications.

Some of the more common AI-based applications in the cloud include:

IoT – Cloud architectures and services that power IoT can store and process data generated by AI platforms on IoT devices.

Chatbots – Chatbots are ubiquitous AI-based software that use natural language processing (NLP) to carry out conversations with users – a boon for customer service in the age of instant gratification. Cloud platforms store and process the data captured by chatbots and cloud services connect them to the appropriate applications for further processing. Customer data is also fed back into the chatbot application that resides in the cloud.

Business Intelligence – BI is another mainstream application where AI is used to gather data on the market, target audience, and competitors of customers. The cloud again facilitates the storage and transfer of data while the AI runs it through predictive analytics models.

AI as a Service (AIaaS) – Public cloud vendors now offer AI outsourcing services, allowing companies to test out software and ML algorithms without risking their primary infrastructure. They can deploy off-the-shelf AI applications at a fraction of the cost of in-house AI with significant CAPEX savings.

Cognitive cloud computing – Cognitive computing is the use of AI models to replicate and simulate human thought processes in complex situations. Players such as IBM and Google have built cognitive cloud platforms that provide cognitive insights-as-a-service to enterprises and facilitate the application of this technology in finance, retail, healthcare, and other industries.

Advantages of Deploying AI in Cloud Environments

AI is the proverbial cherry on the cloud cake. And also the frosting, ganache, strawberries, and sprinkles combined. Here's why AI and cloud form a winning team:

Cost savings – Traditionally ML-based models ran on expensive machines with multiple GPUs in enterprise datacenters. With advances in virtualization in both public and private clouds, the cost of building, testing, and deploying these models has come down drastically. It has leveled the playing field for many small-to-medium businesses.

"I can start up my AI skillsets with just a credit card these days. Back when I first got out of college, we were building things that literally cost $100 million of data center space just to get simple questions answered," said David Linthicum, Chief Cloud Strategy Officer at Deloitte Consulting.

Productivity – AI-based algorithms required significant admin time and effort in terms of building testing and production environments, software management, and provisioning hardware resources for compute operations and storage. A centrally managed hybrid cloud or a public cloud does away with this, leaving IT staff to focus on non-repetitive tasks.

Automation – AI is also being embedded right into the cloud infrastructure to help automate routine processes and streamline workloads. In a hybrid cloud environment, AI tools can be used to monitor, manage, and self-heal individual public and private cloud components.

Analytics – Data residing in most cloud workloads needs to be analyzed for more insights. AI-based models make it easy to mine this data in real time and develop native analytics and dashboards for each of these applications.

Data management – AI helps boost cloud workloads in customer service, marketing, ERP, and supply chain management by processing and generating data in real time. For example, AI tools embedded in Dataflow, the streaming analytics platform in Google Cloud, can enable functions as varied as programmatic bidding in media advertising, fraud prevention in financial services, threat detection in IT security, and personalized shopping recommendations in retail.

Better SaaS tools – Perhaps the most obvious and popular use of AI-based algorithms is their integration in mainstream SaaS tools to help these deliver more functionality and value to end users. For example, Salesforce added "Einstein," an AI-based algorithm to its flagship CRM system to help customers make sense of the immense volumes of data generated, find patterns in this data, and derive actionable insights to improve their sales strategies.

This is but one example in a landscape of literally hundreds of AI-enabled SaaS tools.

Challenges in Deploying AI in Cloud Environments

Merging AI and the cloud isn't always cakes and ale. The main concerns are data privacy and connectivity.

Data privacy

Connectivity

15. Cloud Computing Overview

Cloud Computing provides us means of accessing the applications as utilities over the Internet. It allows us to create, configure, and customize the applications online.

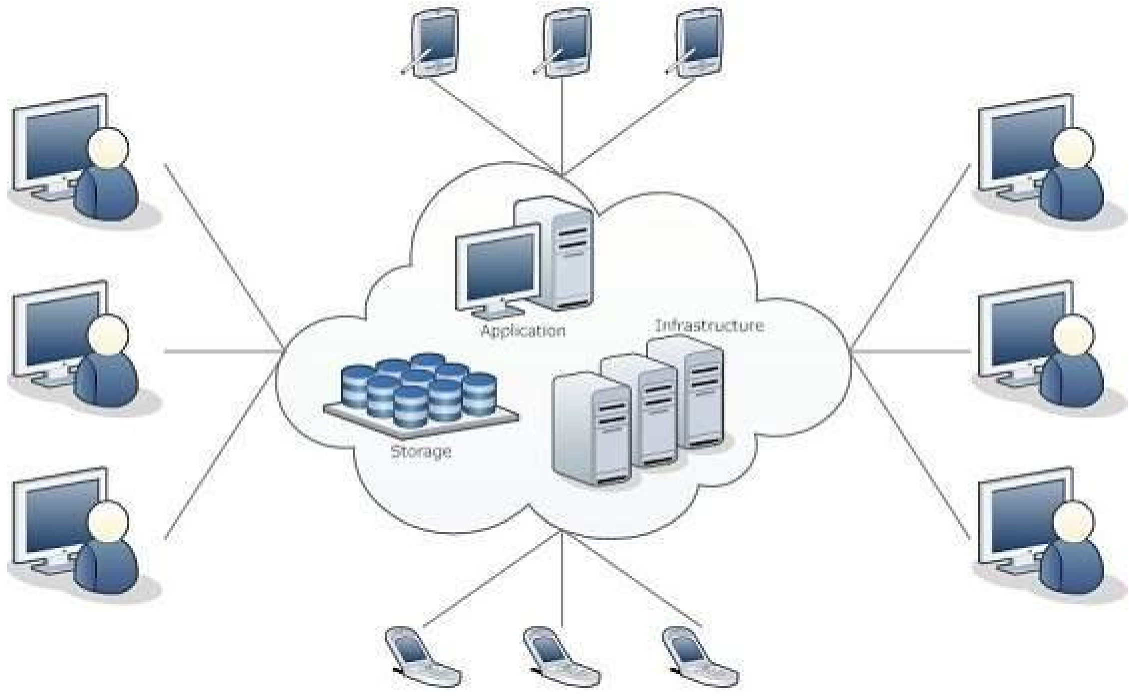
What is Cloud?

The term **Cloud** refers to a **Network** or **Internet.** In other words, we can say that Cloud is something, which is present at remote location. Cloud can provide services over public and private networks, i.e., WAN, LAN or VPN.

Applications such as e-mail, web conferencing, customer relationship management (CRM) execute on cloud.

What is Cloud Computing?

Cloud Computing refers to **manipulating, configuring,** and **accessing** the hardware and software resources remotely. It offers online data storage, infrastructure, and application.



Cloud computing offers **platform independency,** as the software is not required to be installed locally on the PC. Hence, the Cloud Computing is making our business applications **mobile** and **collaborative.**

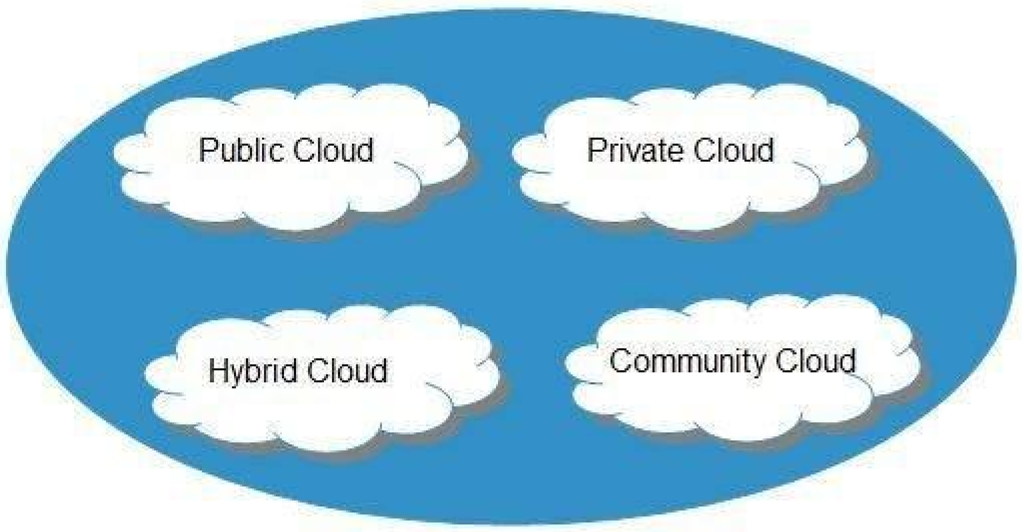
Basic Concepts

There are certain services and models working behind the scene making the cloud computing feasible and accessible to end users. Following are the working models for cloud computing:

- Deployment Models
- Service Models

Deployment Models

Deployment models define the type of access to the cloud, i.e., how the cloud is located? Cloud can have any of the four types of access: Public, Private, Hybrid, and Community.



Public Cloud

The **public cloud** allows systems and services to be easily accessible to the general public. Public cloud may be less secure because of its openness.

Private Cloud

The **private cloud** allows systems and services to be accessible within an organization. It is more secured because of its private nature.

Community Cloud

The **community cloud** allows systems and services to be accessible by a group of organizations.

Hybrid Cloud

The **hybrid cloud** is a mixture of public and private cloud, in which the critical activities are performed using private cloud while the non-critical activities are performed using public cloud.

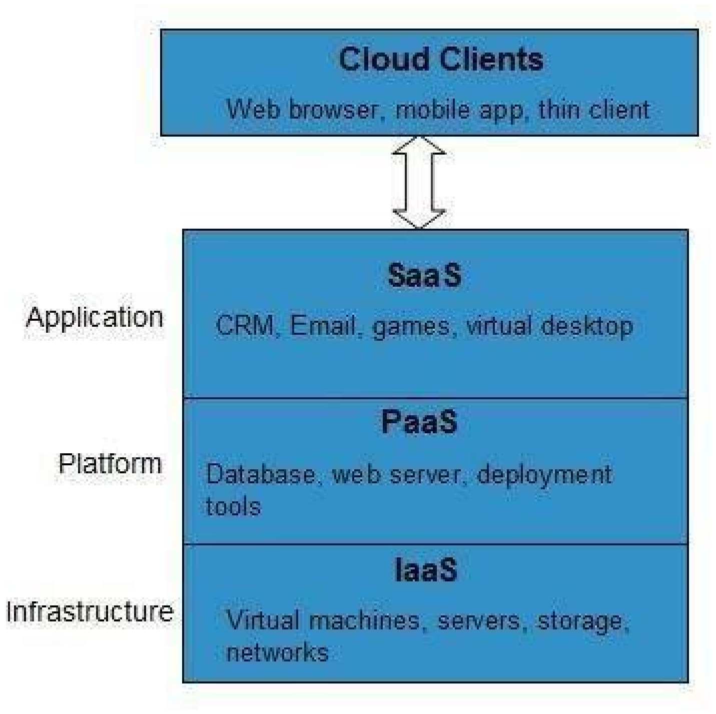
Service Models

Cloud computing is based on service models. These are categorized into three basic service models which are -

- Infrastructure-as–a-Service (IaaS)
- Platform-as-a-Service (PaaS)
- Software-as-a-Service (SaaS)

Anything-as-a-Service (XaaS) is yet another service model, which includes Network-as-a-Service, Business-as-a-Service, Identity-as-a-Service, Database-as-a-Service or Strategy-as-a-Service.

The **Infrastructure-as-a-Service (IaaS)** is the most basic level of service. Each of the service models inherit the security and management mechanism from the underlying model, as shown in the following diagram:


Cloud Clients
Web browser, mobile app, thin client
SaaS
Application
CRM, Email, games, virtual desktop
PaaS
Platform
Database, web server, deployment tools
IaaS
Infrastructure
Virtual machines, servers, storage, networks

Infrastructure-as-a-Service (IaaS)

IaaS provides access to fundamental resources such as physical machines, virtual machines, virtual storage, etc.

Platform-as-a-Service (PaaS)

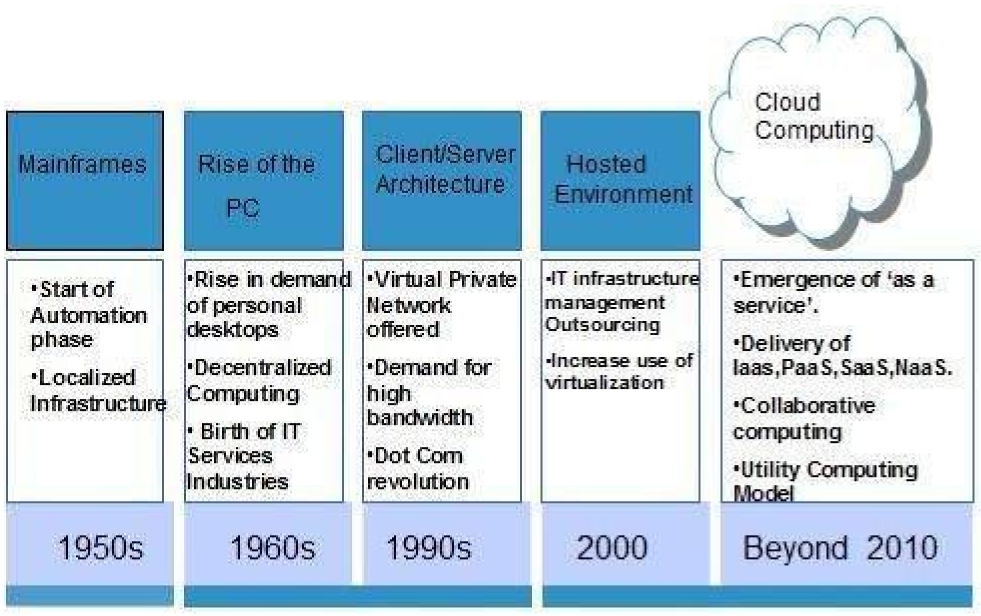
PaaS provides the runtime environment for applications, development and deployment tools, etc.

Software-as-a-Service (SaaS)

SaaS model allows to use software applications as a service to end-users.

History of Cloud Computing

The concept of **Cloud Computing** came into existence in the year 1950 with implementation of mainframe computers, accessible via **thin/static clients.** Since then, cloud computing has been evolved from static clients to dynamic ones and from software to services. The following diagram explains the evolution of cloud computing:


Mainframes
Rise of the PC
Client/Server Architecture
Hosted Environment
Cloud Computing
•Start of Automation phase
•Localized Infrastructure
•Rise in demand of personal desktops
•Decentralized Computing
• Birth of IT Services Industries
•Virtual Private Network offered
•Demand for high bandwidth
•Dot Com revolution
•IT infrastructure management Outsourcing
•Increase use of virtualization
•Emergence of 'as a service'.
•Delivery of Iaas,PaaS,SaaS,NaaS.
•Collaborative computing
•Utility Computing Model
1950s
1960s
1990s
2000
Beyond 2010

16.Cloud Computing Architecture

Cloud Computing , which is one of the demanding technology of the current time and which is giving a new shape to every organization by providing on demand virtualized services/resources. Starting from small to medium and medium to large, every organization use cloud computing services for storing information and accessing it from anywhere and any time only with the help of internet. In this article, we will know more about the internal architecture of cloud computing.

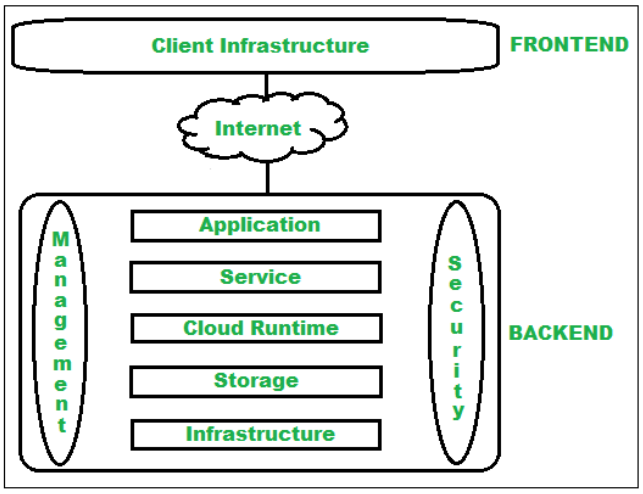
Transparency, scalability, security and intelligent monitoring are some of the most important constraints which every cloud infrastructure should experience. Current research on other important constraints is helping cloud computing system to come up with new features and strategies with a great capability of providing more advanced cloud solutions.

Cloud Computing Architecture :

The cloud architecture is divided into 2 parts i.e.

1. Frontend
2. Backend

The below figure represents an internal architectural view of cloud computing.



Architecture of Cloud Computing

Architecture of cloud computing is the combination of both SOA (Service Oriented Architecture) and EDA (Event Driven Architecture). Client infrastructure, application, service, runtime cloud, storage, infrastructure, management and security all these are the components of cloud computing architecture.

1. Frontend :

Frontend of the cloud architecture refers to the client side of cloud computing system. Means it contains all the user interfaces and applications which are used by the client to access the cloud computing services/resources. For example, use of a web browser to access the cloud platform.

- **Client Infrastructure** – Client Infrastructure is a part of the frontend component. It contains the applications and user interfaces which are required to access the cloud platform.
- In other words, it provides a GUI(Graphical User Interface) to interact with the cloud.

2. Backend :

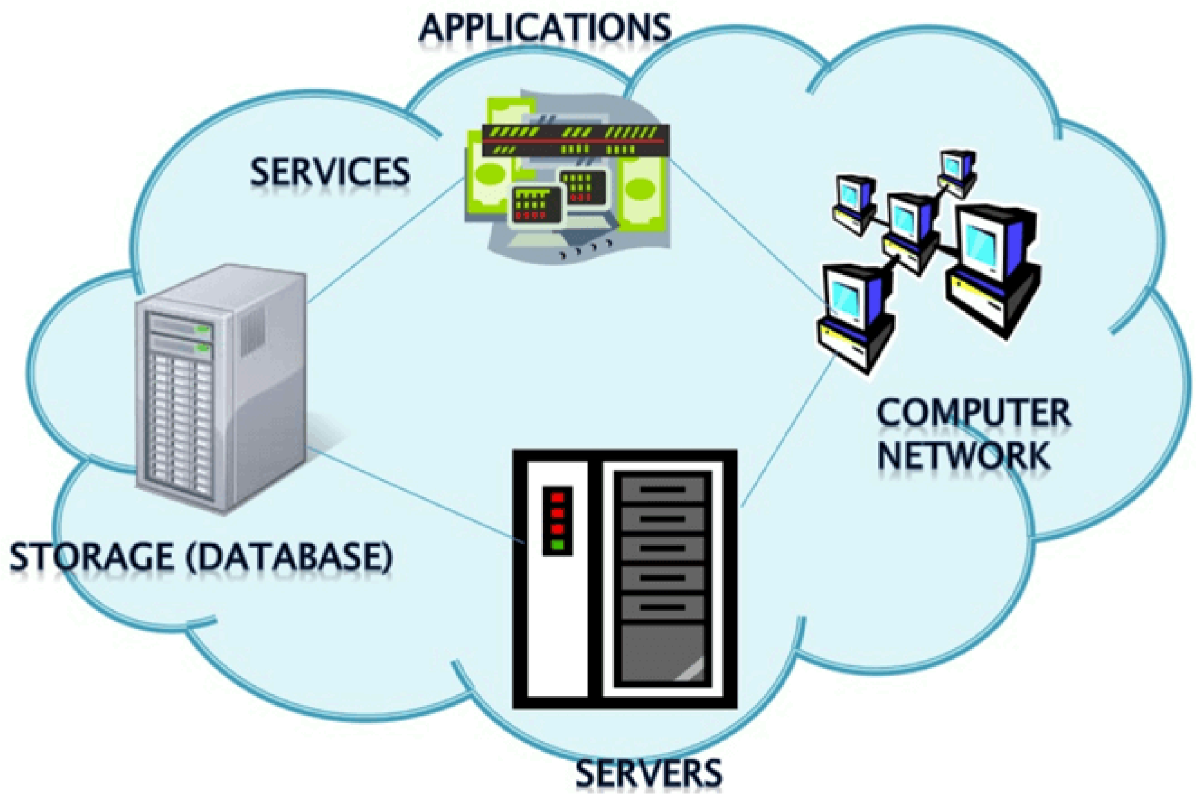
Backend refers to the cloud itself which is used by the service provider. It contains the resources as well as manages the resources and provides security mechanisms. Along with this, it includes huge storage, virtual applications, virtual machines, traffic control mechanisms, deployment models, etc.

1. **Application –**
   Application in backend refers to a software or platform to which client accesses. Means it provides the service in backend as per the client requirement.
2. **Service –**
   Service in backend refers to the major three types of cloud based services like SaaS, PaaS and IaaS. Also manages which type of service the user accesses.
3. **Runtime Cloud-**
   Runtime cloud in backend provides the execution and Runtime platform/environment to the Virtual machine.
4. **Storage –**
   Storage in backend provides flexible and scalable storage service and management of stored data.
5. **Infrastructure –**
   Cloud Infrastructure in backend refers to the hardware and software components of cloud like it includes servers, storage, network devices, virtualization software etc.
6. **Management –**
   Management in backend refers to management of backend components like application, service, runtime cloud, storage, infrastructure, and other security mechanisms etc.
7. **Security –**
   Security in backend refers to implementation of different security mechanisms in the backend for secure cloud resources, systems, files, and infrastructure to end-users.
8. **Internet –**
   Internet connection acts as the medium or a bridge between frontend and backend and establishes the interaction and communication between frontend and backend.

17.Advantages and Disadvantages of Cloud Computing

Advantages of Cloud Computing

Here, we will learn what are the benefits of Cloud Computing in your organization:



Cloud Computing Overview

Cost Savings

Cost saving is one of the biggest Cloud Computing benefits. It helps you to save substantial capital cost as it does not need any physical hardware investments. Also, you do not need trained personnel to maintain the hardware. The buying and managing of equipment is done by the cloud service provider.

Strategic edge

Cloud computing offers a competitive edge over your competitors. It is one of the best advantages of Cloud services that helps you to access the latest applications any time without spending your time and money on installations.

High Speed

Cloud computing allows you to deploy your service quickly in fewer clicks. This faster deployment allows you to get the resources required for your system within fewer minutes.

Back-up and restore data

Once the data is stored in a Cloud, it is easier to get the back-up and recovery of that, which is otherwise very time taking process on-premise.

Automatic Software Integration

In the cloud, software integration is something that occurs automatically. Therefore, you don’t need to take additional efforts to customize and integrate your applications as per your preferences.

Reliability

Reliability is one of the biggest benefits of Cloud hosting. You can always get instantly updated about the changes.

Mobility

Employees who are working on the premises or at the remote locations can easily access all the could services. All they need is an Internet connectivity.

Unlimited storage capacity

The cloud offers almost limitless storage capacity. At any time you can quickly expand your storage capacity with very nominal monthly fees.

Collaboration

The cloud computing platform helps employees who are located in different geographies to collaborate in a highly convenient and secure manner.

Quick Deployment

Last but not least, cloud computing gives you the advantage of rapid deployment. So, when you decide to use the cloud, your entire system can be fully functional in very few minutes. Although, the amount of time taken depends on what kind of technologies are used in your business.

Other Important Benefits of Cloud Computing

Apart from the above, some other Cloud Computing advantages are:

- On-Demand Self-service

- Multi-tenancy
- Offers Resilient Computing
- Fast and effective virtualization
- Provide you low-cost software
- Offers advanced online security
- Location and Device Independence
- Always available, and scales automatically to adjust to the increase in demand
- Allows pay-per-use
- Web-based control & interfaces
- API Access available.

Disadvantages of Cloud Computing

Here, are significant challenges of using Cloud Computing:

Performance Can Vary

When you are working in a cloud environment, your application is running on the server which simultaneously provides resources to other businesses. Any greedy behavior or DDOS attack on your tenant could affect the performance of your shared resource.

Technical Issues

Cloud technology is always prone to an outage and other technical issues. Even, the best cloud service provider companies may face this type of trouble despite maintaining high standards of maintenance.

Security Threat in the Cloud

Another drawback while working with cloud computing services is security risk. Before adopting cloud technology, you should be well aware of the fact that you will be sharing all your company's sensitive information to a third-party cloud computing service provider. Hackers might access this information.

Downtime

Downtime should also be considered while working with cloud computing. That's because your cloud provider may face power loss, low internet connectivity, service maintenance, etc.

Internet Connectivity

Good Internet connectivity is a must in cloud computing. You can't access cloud without an internet connection. Moreover, you don't have any other way to gather data from the cloud.

Lower Bandwidth

Many cloud storage service providers limit bandwidth usage of their users. So, in case if your organization surpasses the given allowance, the additional charges could be significantly costly

Lacks of Support

Cloud Computing companies fail to provide proper support to the customers. Moreover, they want their user to depend on FAQs or online help, which can be a tedious job for non-technical persons.

Conclusion:

Despite all the Cloud Computing advantages and disadvantages, we can't deny the fact that Cloud Computing is the fastest growing part of network-based computing. It offers a great advantage to customers of all sizes: simple users, developers, enterprises and all types of organizations. So, this technology here to stay for a long time.

18.Business Benefits of Cloud Computing

The following 14 benefits are the main reasons cloud computing is being used by most companies.

1. Efficiency/Cost Reduction

With the aid of cloud computing, users can get applications to market quickly without worrying about underlying infrastructure costs or maintenance. Cloud computing helps to lower your operating costs, and also helps organizations pace their investments so they avoid big up-front capital expenses and pay monthly as their business scales.

2. Data Security

The attempts to breach the data of any successful organization is becoming common and cloud computing servers are becoming easy targets for the data breach.

The cloud data has provided a security solution that helps in protecting sensitive information and transaction of your business. This helps the data to stay secure against third party or violation of the data transmitted.

3. Flexibility

The flexibility of the cloud has made employees work in organizations more flexible as they can now complete their tasks in and out of the workplace. They can get any and every information they need on the web browser from the comfort of their abode. No restrictions whatsoever!

4. Mobility

Cloud Mobility has made it easier to connect better and reach more people and information easier, faster and even more securely. It is simply amazing! So, if you're stranded maybe while looking for that piece of information to complete your task, the mobility of the cloud has made it easy for you to communicate, get information and be on the go!

5. Scalability

This is one of the key features and benefits of cloud computing. Scalability enables you to increase or decrease the size or power of the IT solution of your business very easily and as fast as possible.

6. Quality Control

When the output of a business, in terms of quality is poor, it would affect the business very negatively. A cloud-based system ensures all documents are stored in one place and in a single format. Quality control ensures that despite the number of people that open your content, it remains consistent and free of any form of error

7. Automatic Software Updates

All cloud-based applications have been designed to automatically refresh and update themselves instead of waiting for computer specialists to be on ground and keep refreshing every minute.

Even if they are on ground, how many times would they refresh a day? That will be human abuse. These apps update themselves and you do not need to wait as waiting could be very annoying especially if you do not have the time or energy.

8. Easy Manageability

This cloud-based system has been designed to be managed by the service provider. So only little maintenance is needed from specialists. The service provider agreement ensures guaranteed and timely delivery, and maintenance of your business infrastructure.

9. Speed

Speed is another great feature of cloud computing. Within a few seconds, you can get any information you so desire from the comfort of your home. All it takes is just keywords entered into the search button and the search is on.

10. Cloud Analytics

Nothing is better than an information that is well organized and well analyzed. Analysis is a very important part of business as it is the strategic and statistical breakdown of your business operations. Cloud analytics can make most businesses effectively scale the storing, processing, and leveraging of data through insights that reflect changing market conditions.

11. Increased Collaboration

What is the aim of business if you cannot work together with people? Cloud computing gives room for increased working together of people in a business. Some organizations even provide cloud based social platform's where employees can meet, discuss and know what's up.

12. Disaster Recovery.

This is like a backup procedure for all information and facts collected. No matter how perfectly built or strong your system is, a single crash can delete all information if it is not backed up. Thus, it is important to back up data so that when a crash happens, within minutes, all information will be restored, and work can start again in earnest.

13. Remote Working

This is a whole lot similar to flexibility. You can be at home and be very useful to your business. It boosts productivity as people love to be working in comfort as long as there is a very stable internet connection.

14. IoT Ready

IOT (Internet of Things) can generate massive amounts of data, and cloud computing provides a pathway for this data to travel across. By allowing a lot of developers to store and access data remotely, developers can access data immediately and work on projects without delay. The IOT (Internet of Things) has also helped developers to remotely store and access data immediately and work without delay.

19.Differences between Cluster and Cloud Computing

Difference between Cloud Computing and Cluster Computing :

Serial Number	Category	Cloud Computing	Cluster Computing
1.	Goal	Providing on demand IT resources and services.	Performing a complex task in a modular approach.

2.	Resource Sharing	Specific assigned resources are not shareable.	Specific assigned resources are not shareable.
3.	Resource type	In cloud computing there is heterogeneous resource type.	In Cluster Computing there is homogeneous resource type.
4.	Virtualization	Virtualization hardware and software resources.	No virtualization resources.
5.	Security	Security through isolation can be achieved.	Security through node credential can be achieved.
6.	Initial Cost	Initial capital cost for setup is very low.	Initial capital cost for setup is very high.
7.	Security Requirement	Very low	Very high
8.	Maintenance	Requires low maintenance.	Requires little more maintenance.
9.	Hardware	No hardware requirement physically.	More hardware requirement physically.
10.	Node OS	Multiple OS runs in VM	Windows, Linux
11.	User Management	User management is centralized or decentralized to vendor/third party.	User management is centralized.
12.	Scalability	Allowed	Limited
13.	Architecture	In Cloud Computing User chosen architecture.	In Cluster Computing Cluster oriented architecture
14.	Characteristic	Dynamic computing infrastructure/resources/services	Tightly coupled systems/resources

15.	Software Dependent	In cloud computing application domain independent software.	In cluster computing application domain dependent software.
16.	Example	Dropbox, Gmail	Sony PlayStation clusters

20.Business Drivers for Adopting Cloud Computing

There are several benefits of adopting the cloud in your business for essential features like document management. Here are some of the main reasons businesses have identified this as an important change.

1. Cost Savings with Cloud Storage

Whenever you make a change in your business, it needs to make sense for your bottom line. Moving from on-site storage to cloud-based storage is one of the major reasons businesses are eager to make the switch. In fact, the savings from making the change have been estimated at 37%. Why are these savings so significant? It comes down to hardware maintenance and updates.

When you move to cloud-based computing, you are utilizing the infrastructure of the cloud provider rather than your own infrastructure. This means that the cost of maintenance, updating equipment, and managing software on the devices becomes the responsibility of the service provider.

2. Cloud Storage Provides Convenience

Businesses are increasingly becoming more mobile as technology enables them to do so. Of course, employees still have to be able to access and share files when they are working outside of the office. With cloud-based storage, users can easily access the files they need wherever they are located. This enables your employees to focus on the tasks that can actually drive growth in your business like meeting face-to-face with clients or prospecting for new business.

3. Flexibility and Scalability

As your business grows, you need your computing services to grow with it. If you are managing your own equipment, this means increased cost as you install new infrastructure and more costs to maintain that equipment. With cloud-based computing, you can simply purchase more storage or computing power to meet your needs and, best of all, have instant access to that expanded capability.

4. Cutting-Edge Security

It's hard not to turn on the daily news and see a story about the latest major data breach. While cloud-based computing isn't 100 percent secure, your business does get the benefit of regular security updates. Most importantly, the cost and management of the new security doesn't fall in your lap. Many cloud providers are also implementing artificial intelligence to help identify and track down the latest threats to provide even greater security.

5. Suited to Your Unique Needs and Budget

Many businesses experience increased periods of work. Retailers, for example, will see a major uptick in activity around the holiday season. With on-premise network and storage infrastructure, you don't have the ability to increase or decrease the power of your infrastructure as needed. Rather, you have to maintain the maximum capabilities needed even during slow times.

With SaaS cloud-based computing, your needs can dictate what you pay for. Increase your capabilities during busy seasons and save money during slow seasons.

6. Rapid Disaster Recovery

A fire or flood in your business could be devastating if you store data on-premise. Not only do you need to recover data, you may need to replace infrastructure at a very high cost. This can lead to prolonged downtime, increased expenses, and lost revenue.

With cloud-based computing and cloud-based storage, a disaster in your business doesn't mean you have to go completely dark. Your data is stored and backed up across several data storage centers. Even a disaster at one storage center can be overcome quickly with redundancy thanks to other storage centers and backups.

7. Improve Collaboration Among Teams

If your business requires multiple teams to collaborate on projects or you have teams in multiple locations, collaboration is vital to the success of your business. With cloud-based storage, teams in different locations can easily share and collaborate on documents seamlessly. There's no need to install additional network infrastructure to connect your on-premise equipment and storage with off-premise teams.

8. Enable More Efficient Document Storage and Review

With advanced capture and cloud storage, documents can instantly be analyzed and stored in a way that makes it easy for your business to access and review documents. This can be incredibly handy if you are in the process of going through an audit.

21.Unit 6: Overview of Distributed and Parallel Computing

Both parallel and distributed computing have been around for a long time and both have contributed greatly to the improvement of computing processes. However, they have key differences in their primary function.
Parallel computing, also known as parallel processing, speeds up a computational task by dividing it into smaller jobs across multiple processors inside one computer. Distributed computing, on the other hand, uses a distributed system, such as the internet, to increase the available computing power and enable larger, more complex tasks to be executed across multiple machines.

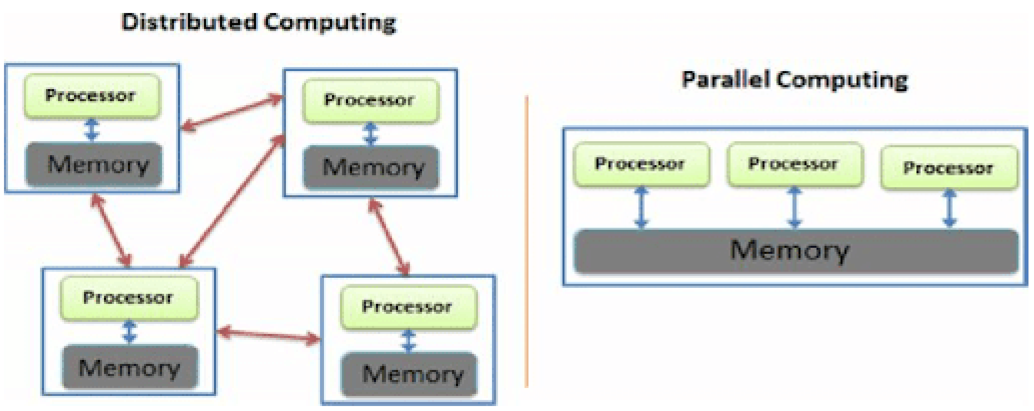


Figure 1: A distributed computing system compared to a parallel computing system.
Source: ResearchGate

22. A. Distributed Computing Overview,

What is a Distributed System?

A distributed system is a collection of multiple physically separated servers and data storage that reside in different systems worldwide. These components can collaborate, communicate, and work together to achieve the same objective, giving an illusion of being a single, unified system with powerful computing capabilities.

A distributed computing server, databases, software applications, and file storage systems can all be considered distributed systems.

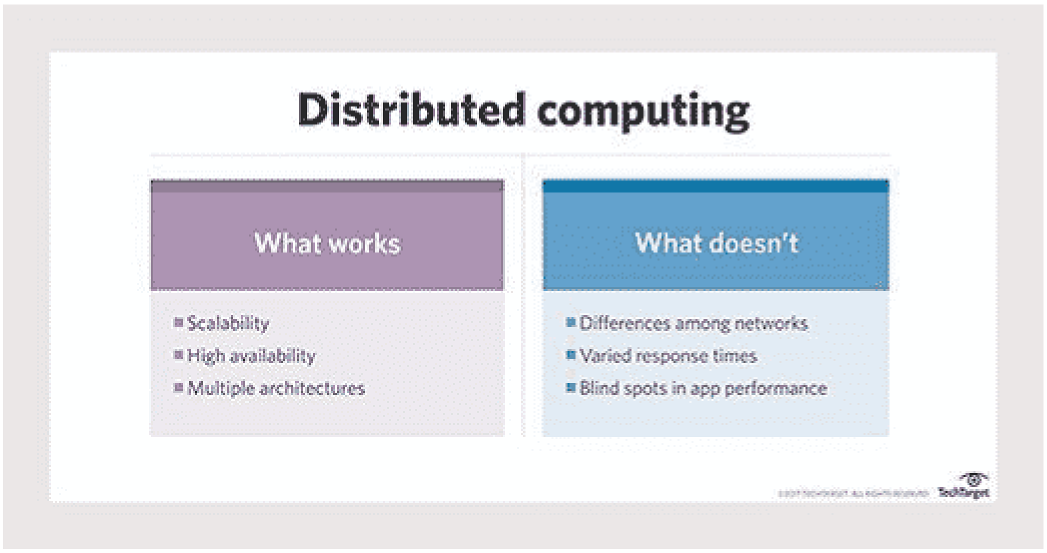
Examples of Distributed Systems

- The internet (World Wide Web) itself.
- Telecommunication networks with multiple antennas, amplifiers, and other networking devices appear as a single system to end-users.

Distributed Computing Definition: What is Distributed Cloud?

In a distributed cloud, the public cloud infrastructure utilizes multiple locations and data centers to store and run the software applications and services. With this implementation, distributed clouds are more efficient and performance-driven.

A distributed cloud computing architecture also called distributed computing architecture, is made up of distributed systems and clouds.



Distributed computing

What works	What doesn't
Scalability	Differences among networks
High availability	Varied response times
Multiple architectures	Blind spots in app performance

Source

Examples of Distributed Computing

- Content Delivery Networks (CDNs) utilize geographically separated regions to store data locally in order to serve end-users faster.
- Ridge Edge Platform

How Does Distributed Computing Work?

Distributed computing connects hardware and software resources to do many things, including:

- Work in collaboration to achieve a single goal through optional **resource sharing**;
- Manage **access rights** per the authority level of users;
- Keep resources, e.g., distributed computing software, **open** for further development;
- Achieve **concurrency** that lets multiple machines work on the same process;
- Ensure all computing resources are **scalable** and operate faster when multiple machines work together;
- Detect and handle errors in connected components of the distributed network so that the network doesn't fail and stays **fault-tolerant**.

Advanced distributed systems have automated processes and APIs to help them perform better.

From the customization perspective, distributed clouds are a boon for businesses. Cloud service providers can connect on-premises systems to the cloud computing stack so that enterprises can transform their entire IT infrastructure without discarding old setups. Instead, they can extend existing infrastructure through comparatively fewer modifications.

The cloud service provider controls the application upgrades, security, reliability, adherence to standards, governance, and disaster recovery mechanism for the distributed infrastructure.

What Are the Advantages of Distributed Cloud Computing?

According to Gartner, distributed computing systems are becoming a primary service that all cloud services providers offer to their clients. Why? Because the advantages of distributed cloud computing are extraordinary. Here is a quick list:

Ultimate Scalability

All nodes or components of the distributed network are independent computers. Together, they form a distributed computing cluster. You can easily add or remove systems from the network without resource straining or downtime. Scaling with distributed computing services providers is easy.

Improved Fault Tolerance

Distributed systems form a unified network and communicate well. At the same time, the architecture allows any node to enter or exit at any time. As a result, fault-tolerant distributed systems have a higher degree of reliability.

Boosted Performance and Agility

Distributed clouds allow multiple machines to work on the same process, improving the performance of such systems by a factor of two or more. As a result of this load balancing, processing speed and cost-effectiveness of operations can improve with distributed systems.

Lower Latency

As resources are globally present, businesses can select cloud-based servers near end-users and speed up request processing. Companies reap the benefit of edge computing's low latency with the convenience of a unified public cloud.

Helpful in Compliance Implementation

Whether there is industry compliance or regional compliance, distributed cloud infrastructure helps businesses use local or country-based resources in different geographies. This way, they can easily comply with varying data privacy rules, such as GDPR in Europe or CCPA in California.

If you want to learn more about the advantages of Distributed Computing, you should read our article on the benefits of Distributed Computing.

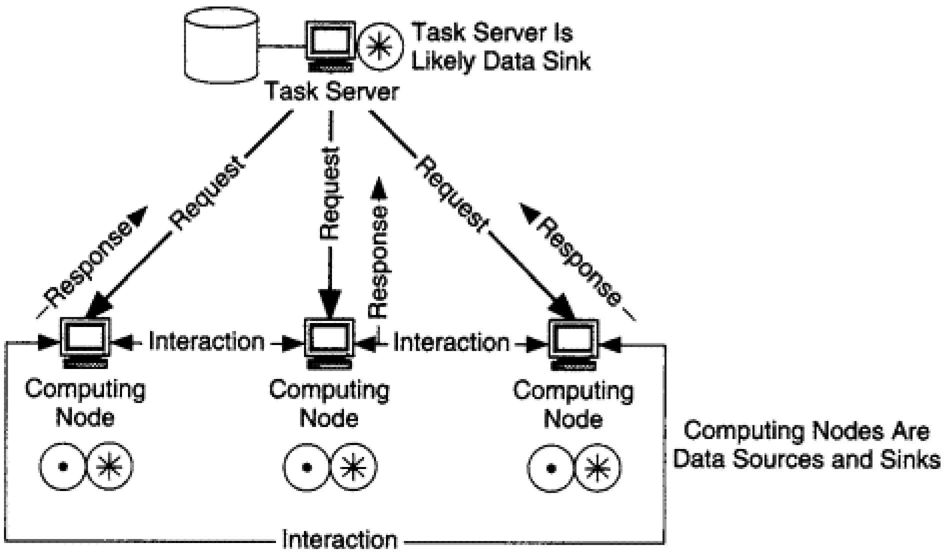
Four Types of Distributed Systems

Under the umbrella of distributed systems, there are a few different architectures. Broadly, we can divide distributed cloud systems into four models:

Client-Server Model

In this model, the client fetches data from the server directly then formats the data and renders it for the end-user. To modify this data, end-users can directly submit their edits back to the server.

For example, companies like Amazon that store customer information. When a customer updates their address or phone number, the client sends this to the server, where the server updates the information in the database.



Three-Tier Model

The three-tier model introduces an additional tier between client and server — the agent tier.

This middle tier holds the client data, releasing the client from the burden of managing its own information. The client can access its data through a web application, typically. Through this, the client application's and the user's work is reduced and automated easily.

For example, a cloud storage space with the ability to store your files and a document editor. Such a storage solution can make your file available anywhere for you through the internet, saving you from managing data on your local machine.

Multi-Tier Model

Enterprises need business logic to interact with various backend data tiers and frontend presentation tiers. This logic sends requests to multiple enterprise network services easily. That's why large organizations prefer the n-tier or multi-tier distributed computing model.

For example, an enterprise network with n-tiers that collaborate when a user publishes a social media post to multiple platforms. The post itself goes from data tier to presentation tier.

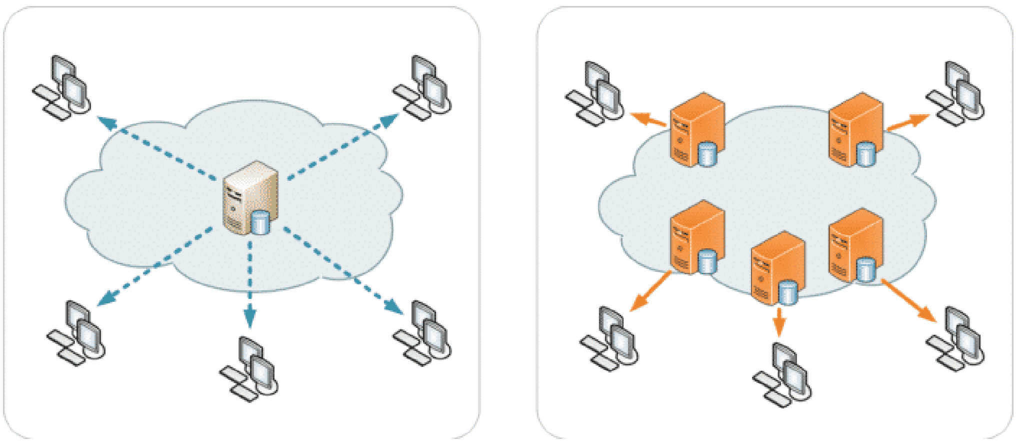
Peer-to-Peer Model

Unlike the hierarchical client and server model, this model comprises peers. Each peer can act as a client or server, depending upon the request it is processing. These peers share their computing power, decision-making power, and capabilities to work better in collaboration.

For example, blockchain nodes collaboratively work to make decisions regarding adding, deleting, and updating data in the network.

Applications of Distributed Computing

CDNs



CDNs place their resources in various locations and allow users to access the nearest copy to fulfill their requests faster. Industries like streaming and video surveillance see maximum benefits from such deployments. If a customer in Seattle clicks a link to a video, the distributed network funnels the request to a local CDN in Washington, allowing the customer to load and watch the video faster.

Real-time or Performance-driven Systems

As real-time applications (the ones that process data in a time-critical manner) must work faster through efficient data fetching, distributed machines greatly help such systems.

Multiplayer games with heavy graphics data (e.g., PUBG and Fortnite), applications with payment options, and torrenting apps are a few examples of real-time applications where distributing cloud can improve user experience.

Distributed Systems and Cloud Computing

Distributed systems and cloud computing are a perfect match that powers efficient networks and makes them fault-tolerant. From storage to operations, distributed cloud services fulfill all of your business needs.

Distributed Computing with Ridge

Using the distributed cloud platform by Ridge, companies can build their very own, customized distributed systems that have the agility of edge computing and power of distributed computing.

Ridge offers managed Kubernetes clusters, container orchestration, and object storage services for advanced implementations. Ridge Cloud takes advantage of the economies of locality and distribution.

As an alternative to the traditional public cloud model, Ridge Cloud enables application owners to utilize a global network of service providers instead of relying on the availability of computing resources in a specific location. And by facilitating interoperability with existing infrastructure, empowers enterprises to deploy and infinitely scale applications anywhere they need.

22.B. Distributed Computing Framework and Architecture

Architecture of Distributed Systems

Cloud-based software, the backbone of distributed systems, is a complicated network of servers that anyone with an internet connection can access. In a distributed system, components and connectors arrange themselves in a way that eases communication. Components are modules with well-defined interfaces that can be replaced or reused. Similarly, connectors are communication links between modules that mediate coordination or cooperation among components.

A distributed system is broadly divided into two essential concepts — software architecture (further divided into layered architecture, object-based architecture, data-centered architecture, and event-based architecture) and system architecture (further divided into client-server architecture and peer-to-peer architecture).

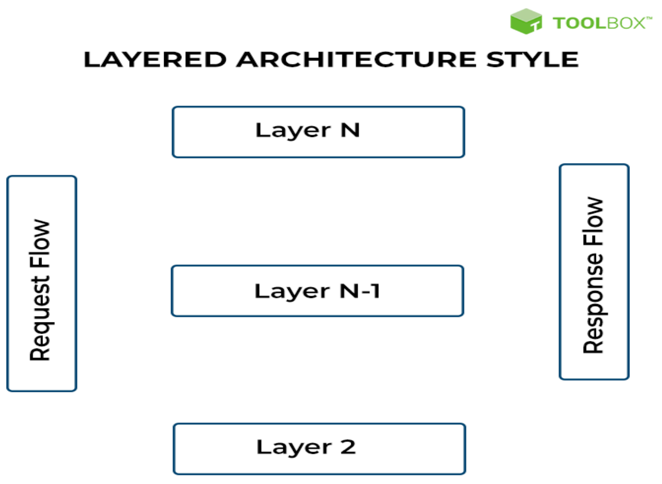
Let's understand each of these architecture systems in detail:

1. Software architecture

Software architecture is the logical organization of software components and their interaction with other structures. It is at a lower level than system architecture and focuses entirely on components; e.g., the web front end of an ecommerce system is a component. The four main architectural styles of distributed systems in software components entail:

i) Layered architecture

Layered architecture provides a modular approach to software. By separating each component, it is more efficient. For example, the open systems interconnection (OSI) model uses a layered architecture for better results. It does this by contacting layers in sequence, which allows it to reach its goal. In some instances, the implementation of layered architecture is in cross-layer coordination. Under cross-layer, the interactions can skip any adjacent layer until it fulfills the request and provides better performance results.

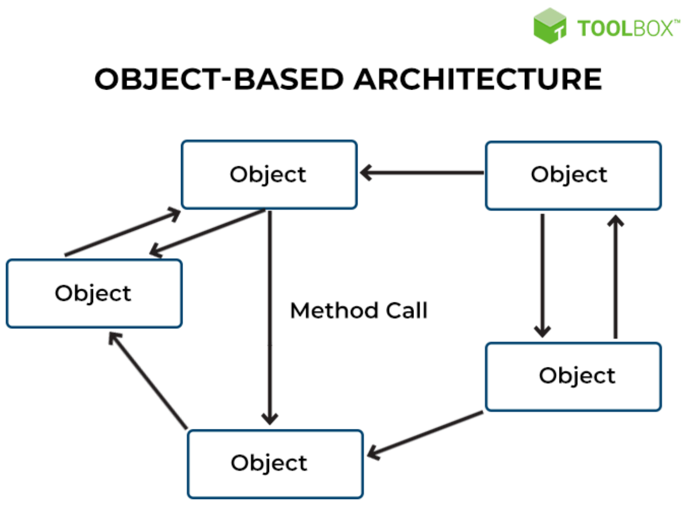


Layered Architecture

Layered architecture is a type of software that separates components into units. A request goes from the top down, and the response goes from the bottom up. The advantage of layered architecture is that it keeps things orderly and modifies each layer independently without affecting the rest of the system.

ii) Object-based architecture

Object-based architecture centers around an arrangement of loosely coupled objects with no specific architecture like layers. Unlike layered architecture, object-based architecture doesn't have to follow any steps in a sequence. Each component is an object, and all the objects can interact through an interface (or connector). Under object-based architecture, such interactions between components can happen through a direct method call.

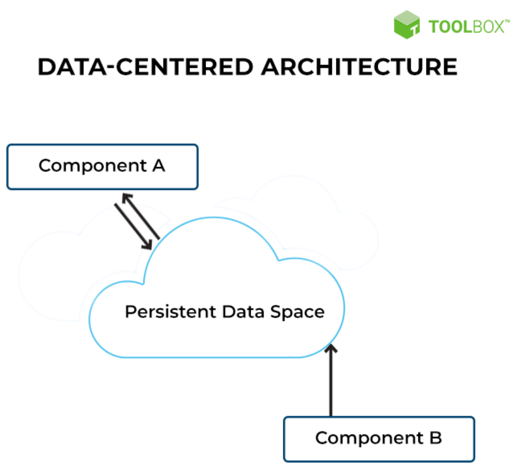


Object-based Architecture

At its core, communication between objects happens through method invocations, often called remote procedure calls (RPC). Popular RPC systems include Java RMI and Web Services and REST API Calls. The primary design consideration of these architectures is that they are less structured. Here, component equals object, and connector equals RPC or RMI.

iii) Data-centered architecture

Data-centered architecture works on a central data repository, either active or passive. Like most producer-consumer scenarios, the producer (business) produces items to the common data store, and the consumer (individual) can request data from it. Sometimes, this central repository can be just a simple database.

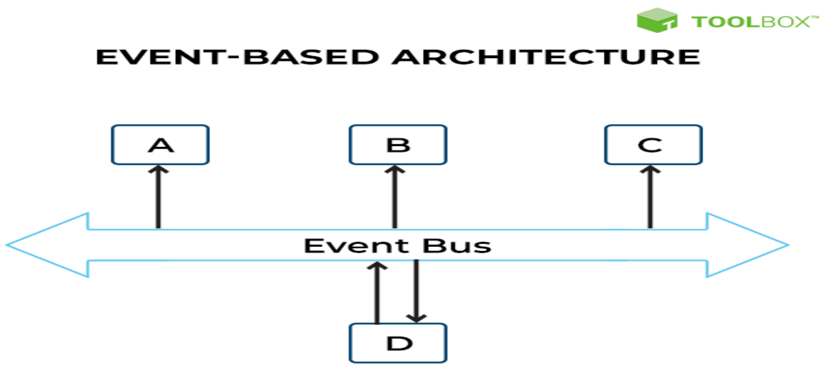


Data-centered Architecture

All communication between objects happens through a data storage system in a data-centered system. It supports its stores' components with a persistent storage space such as an SQL database, and the system stores all the nodes in this data storage.

iv) Event-based architecture

In event-based architecture, the entire communication is through events. When an event occurs, the system gets the notification. This means that anyone who receives this event will also be notified and has access to information. Sometimes, these events are data, and at other times they are URLs to resources. As such, the receiver can process what information they receive and act accordingly.



Event-Based Architecture

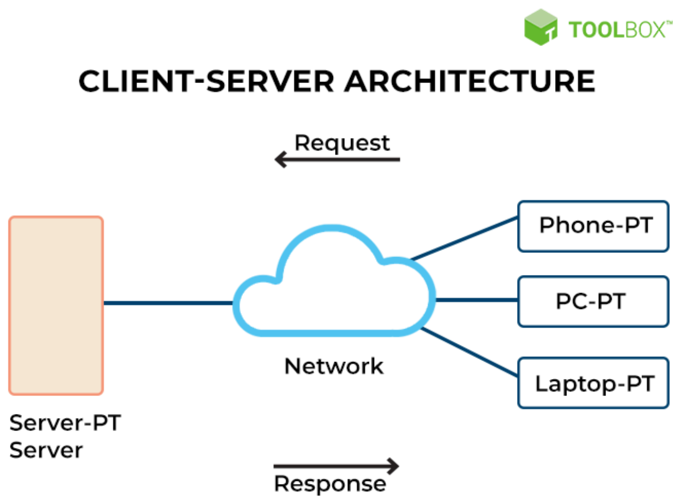
One significant advantage of event-based architecture is that the components are loosely coupled. Eventually, it means that it’s easy to add, remove, and modify them. To better understand this, think of publisher-subscriber systems, enterprise services buses, or akka.io. One advantage of event-based architecture is allowing heterogeneous components to communicate with the bus, regardless of their communication protocols.

2. System architecture

System-level architecture focuses on the entire system and the placement of components of a distributed system across multiple machines. The client-server architecture and peer-to-peer architecture are the two major system-level architectures that hold significance today. An example would be an ecommerce system that contains a service layer, a database, and a web front.

i) Client-server architecture

As the name suggests, client-server architecture consists of a client and a server. The server is where all the work processes are, while the client is where the user interacts with the service and other resources (remote server). The client can then request from the server, and the server will respond accordingly. Typically, only one server handles the remote side; however, using multiple servers ensures total safety.

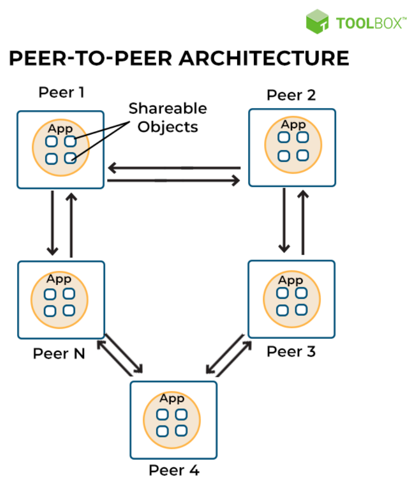


Client-server Architecture

Client-server architecture has one standard design feature: centralized security. Data such as usernames and passwords are stored in a secure database for any server user to have access to this information. This makes it more stable and secure than peer-to-peer. This stability comes from client-server architecture, where the security database can allow resource usage in a more meaningful way. The system is much more stable and secure, even though it isn't as fast as a server. The disadvantages of a distributed system are its single point of failure and not being as scalable as a server.

ii) Peer-to-peer (P2P) architecture

A peer-to-peer network, also called a (P2P) network, works on the concept of no central control in a distributed system. A node can either act as a client or server at any given time once it joins the network. A node that requests something is called a client, and one that provides something is called a server. In general, each node is called a peer.



Peer-to-Peer Architecture

If a new node wishes to provide services, it can do so in two ways. One way is to register with a centralized lookup server, which will then direct the node to the service provider. The other way is for the node to broadcast its service request to every other node in the network, and whichever node responds will provide the requested service.

P2P networks of today have three separate sections:

- **Structured P2P:** The nodes in structured P2P follow a predefined distributed data structure.
- **Unstructured P2P:** The nodes in unstructured P2P randomly select their neighbors.
- **Hybrid P2P:** In a hybrid P2P, some nodes have unique functions appointed to them in an orderly manner.

23. Business Challenges of Distributed Computing.

Distributed Systems: Challenges, Failures

Challenges and Failures of a Distributed System are:

- Heterogeneity

- Scalability
- Openness
- Transparency
- Concurrency
- Security
- Failure Handling

Heterogeneity

Heterogeneity refers to the differences that arise in networks, programming languages, hardware, operating systems and differences in software implementation. For example, there are different hardware devices, tablets, mobile phones, computers, and etc.

Some challenges may present themselves due to heterogeneity. When programs are written in different languages or developers utilize different implementations (data structures, etc.) problems will arise when the computers try to communicate with each other. Thus it is important to have common standards agreed upon and adopted to streamline the process. Additionally, when we consider mobile code - code that can be transferred from one computer to the next - we may encounter some problems if the executables are not specified to accomodate both computers' instructions and specifications.

Scalability

A program is scalable if a program does not need to be redesigned to ensure stability and consistent performance as its workload increases. As such, a program (distributed system in our case) should not have a change in performance regardless of whether it has 10 nodes or 100 nodes.

Openness

The openness of distributed systems refers to the system's extensability and ability to be reimplemented. More specifically, the openness of a distributed system can be measured by three characteristics: interoperability, portability, and extensability as we previously mentioned. Interoperability refers to the system's ability to effectively interchange information between computers by standardization, portability refers to the system's ability to properly function on different operating systems, and extensability allows developers to freely add new features or easily reimplement existing ones without impairing functionality. Additionally, open distributed systems implement open interfaces, which comes with many challenges - in this case having well-defined interfaces may present themselves as challenges.

Transparency

A problem with transparency may arise with distributed systems due to the nature of the system's complexity. In this context, transparency refers to the distributed system's ability to conceal its complexity and give off the apperance of a single system. And when we discuss transparency, we must also discuss to what extent.

Concurrency

This discusses the shared access of resources which must be made available to the correct processes. Problems may arise when multiple processes attempt to access the same resources at the same time, thus steps need to be taken to ensure that any manipulation in the system remains in a stable state;

Security

Security is comprised of three key components: availability, integrity, and confidentiality. In a similar fashion, authentication and authorization are paramount - an entity must be verifiably authenticated as claimed and privileges must be appropriate delegated based on authority.

These concepts are related, availability regards the authenticated and authorized users, integrity protects through encryption and other methods, and confidentiality ensures that resources are not needlessly disclosed or made available.

Security is especially important in distributive systems due to their association with sensitive and private data. Take payment and transactions information, for example.

Failure Handling

Failures, like in any program, are a major problem. However, in distributive systems, with so many processes and users, the consequences of failures are exacerbated. Additionally, many problems will arise due to the nature of distributive systems. Unexpected edge cases may present themselves in which the system is ill-equipped for, but developers must account for. Failures can occur in the software, hardware, and in the network; additionally the failure can be partial, causing some components to function and other to not. However, the most important part in failure handling is recognizing that not every failure can be accounted for. Thus, implementing processes to detect, monitor, and repair systems failures is a core feature in failure handling/ management.

With this article at OpenGenus, you must have a strong idea of Challenges/ Failures in Distributed Systems.

23.B. Business Benefits of Distributed Computing

The list of benefits that Distributed Cloud computing can potentially offer any business is infinite. In the current age of WFH and remote working seeking prominence and taking shape as the primary work model across most industries, the Distributed cloud enables the whole system to work seamlessly by allowing your widely distributed employees to set up their virtual office or working space concerning where at any time point giving your business the facility and the flexibility of staying connected despite the barriers. With the expansion of employees across the globe and scaling of web-enabled devices interlinked to today's business ecosystem (e.g. smartphones, tablets, smartwatches), Distributed cloud computing increases outreach to your essential business data enabling efficient working across, while enjoying the additional advantages of cutting-edge automation of minimal routine tasks and supporting applications aiding in decision making.

So, let's take a look at the compiled list of benefits that your business can leverage by switching to Distributed cloud computing:

1. Reduced IT costs

One of the most prominent advantages of shifting to Distributed Cloud is improved cost efficiency. Shifting to cloud computing cuts down costs in the segment of managing and maintaining your IT department and employees working particularly in that domain. Not only a significant cost reduction takes place on infrastructure but also several teams can be eliminated who were allotted with the primary task to oversee, manage and sustain the process.

Post-implementation, the employee bandwidth saved can be utilized in other spheres that can benefit from it.

You will be able to contain operating costs because:

- It's a one-time investment with no additional expenditure on the installation of additional hardware and software to match multiple upgradations
- Significant reduction in maintenance and payment for expert staff in the particular domain
- Lowered energy consumption over time leading to reduced costs
- Improved functionality leading to fastened time response

2. Network Scalability

As per changing business demands, any organization needs to scale up or down different crucial dimensions such as storage, operations, and technical capacity for improved accommodation of the situation at hand and continue the business process with absolute flexibility. Since different applications operate on different servers, it is feasible to add or remove additional resources concerning requirements without disturbing the original version of the cloud integration across the network. Not only does it cut down the cost and additional effort going into an expensive upgrade but it also helps you to utilize your time better.

Distributed cloud specifically enables a company to create and unfold in any environment with the pre-existing set of equipment and staff, eliminating the need to build new edge locations that burn time, effort, and money.

3. Reduces Latency

Distributed Cloud offers cutting-edge solutions to the problems relating to latency. It makes the services much faster and responsive by bringing critical processing tasks closer to the end-up disposal. This is possible as the intermediary sub-processes such as data processing are executed locally instead of doing the additional task where the data gathered has to be transmitted further to the centralized network servers for analysis, as a result, it fastens the response time by navigating the dominating traffic.

By increasing the proximity of data to its origin and reduced dependency on data transmission back to the center, Distributed cloud hugely improved performance speed. Clearly, for this very reason, it has become the heart of data-driven advancements such as an autonomous vehicle, where time is the most crucial parameter to ensure safety.

4. Greater autonomy and security

With security as one of the primary concerns of today's time, Implementation of the Distributed cloud takes over the computation and storage department making an organization independent over time and self-efficient to handle its essential needs. As a company takes charge of these basic processes. it obtains a higher degree of control and overall protection where it can supervise all the processes at play through the controlling layer. The company has access to everything in one convenient spot, a micro-cloud that is located exterior to the centralized cloud. It has dual benefits, as it in one it ensures security and the same time makes a business more agile overall resulting in better business results and success.

5. Builds resilience

A distributed cloud ecosystem amplifies the business resilience of any organization to unpredictable situations such as system failure that may bring operations to a halt. It is the most cost-effective solution that saves a system crash from affecting other servers inter-linked in the networks. By unit decentralization, the overall compounded effect of failure or pertaining failure risks is reduced and eliminates the need for backend preparation for spare capacity enabling organizations for better adaptation.

6. Easier regulatory compliance

The distributed cloud framework follows the basic principles of constructing services pertaining based on policy-regulated local storage. It thus aids the businesses to adhere to the mandatory practices, as per which organizations should handle data within the given jurisdiction such as nation, region, or business eco-system. In certain other circumstances where the data privacy laws state that any individual's personal information cannot be handled outside the marked jurisdiction, Distributed cloud comes rescue for enabling smooth business operations. It helps in enabling the organizations to process personal information within the user's permitted ecosystem. This aspect has attained relevance and usage across several industries such as telecommunication, healthcare, etc.

7. Fastened Content Delivery

Distributed cloud has the basic operational level close to the users which ultimately improves the output speed of delivery. It is known for top-notch video experience as the content delivery network (CDN) available on the platform improves the streaming video activity by manifold. For applications that utilize the latest advancements such as IoT(Internet of Things), AI (Artificial Intelligence), and others commonly video surveillance and self-driving cars demand faster and efficient data processing which can be executed in real-time. In such cases distributed cloud and edge computing reduce latency to accommodate the need.

Apart from that cloud computing has the provision of diverting the queries posed by a user to the designated server which possesses all the other details and transaction history related to the individual. This significantly brings down the extra load of processing all the requests from a single server machine. As well as it helps in passing on a certain request to the physically closest or the fastest network connection which leads to reduces time and resources that require allocation for catering to the incoming traffic and bottlenecks.

8. Access To Automatic Updates

Distributed clouds have the inbuilt capability attribute to notify about the latest developments and other necessary changes including auto-update. Depending on the service provider, your system periodically gets updated with the latest technology as well as upgrades to the most recent and effective server. Also, it has the facility for seamless software automation which further reduces the back-end efforts and time that one needs to invest otherwise. As a business dealing with large-scale employees from different regions across the globe, distributed cloud boosts their ability to work effectively despite the barriers generating income and results for the company.

Applications of Distributed Computing

Distributed computing has become an **essential basic technology involved in the digitalization** of both our private life and work life. The internet and the services it offers would not be possible if it were not for the client-server architectures of distributed systems. Every Google search involves distributed computing with supplier instances around the world working together to generate matching search results. Google Maps and Google Earth also leverage distributed computing for their services.

Distributed computing methods and architectures are also used in email and conferencing systems, airline and hotel reservation systems as well as libraries and navigation systems. In the working world, the primary applications of this technology include **automation processes** as well as planning, production, and design systems. Social networks, mobile systems, **online banking, and online gaming** (e.g. multiplayer systems) also use efficient distributed systems.

Additional areas of application for distributed computing include e-learning platforms, artificial intelligence, and **e-commerce**. Purchases and orders made in online shops are usually carried out by distributed systems. In meteorology, **sensor and monitoring systems** rely on the computing power of distributed systems to forecast natural disasters. Many digital applications today are based on distributed databases.

Particularly **computationally intensive research projects** that used to require the use of expensive supercomputers (e.g. the Cray computer) can now be conducted with more cost-effective distributed systems. The **volunteer computing project** SETI@home has been setting standards in the field of distributed computing since 1999 and still are today in 2020. Countless networked **home computers belonging to private individuals** have been used to evaluate data from the Arecibo Observatory radio telescope in Puerto Rico and support the University of California, Berkeley in its search for extraterrestrial life.

A unique feature of this project was its **resource-saving approach**. The analysis software only worked during periods when the user’s computer had nothing to do. After the signal was analyzed, the results were sent back to the headquarters in Berkeley.