**Distributed computing**

**1.1 Definition**

A distributed system is a collection of independent entities that cooperate to solve a problem that cannot be individually solved. Distributed systems have been in existence since the start of the universe. From a school of fish to a flock of birds and entire ecosystems of microorganisms, there is communication among mobile intelligent agents in nature. With the widespread proliferation of the Internet and the emerging global village, the notion of distributed computing systems as a useful and widely deployed tool is becoming a reality.

A distributed system is a computing environment in which various components are spread across multiple computers (or other computing devices) on a network. These devices split up the work, coordinating their efforts to complete the job more efficiently than if a single device had been responsible for the task. Distributed systems reduce the risks involved with having a single point of failure, bolstering reliability and fault tolerance. Modern distributed systems are generally designed to be scalable in near real-time and additional computing resources can be added on the fly to increasing performance and further reducing time to completion. Earlier, distributed computing was expensive, complex to configure and difficult to manage But, Software as a Service (SaaS) platforms has offered expanded functionality, distributed computing has become more streamlined and affordable for businesses, large and small, all types of computing jobs be it database management, video games or Software cryptocurrency systems, scientific simulations, blockchain technologies and AI platforms all use Distributed Systems platforms.

**1.2 Characteristics of distributed computing**

For computing systems, a distributed system has been characterized in one of several ways:

* You know you are using one when the crash of a computer you have never heard of prevents you from doing work.
* A collection of computers that do not share common memory or a common physical clock, that communicate by a message passing over a communication network, and where each computer has its own memory and runs its own operating system. Typically, the computers are semi-autonomous and are loosely coupled while they cooperate to address a problem collectively.
* A collection of independent computers that appears to the users of the system as a single coherent computer.
* A term that describes a wide range of computers, from weakly coupled systems such as wide-area networks, to strongly coupled systems such as local area networks, to very strongly coupled systems such as multiprocessor systems.

**1.3 Features of distributed computing**

A distributed system can be characterized as a collection of mostly autonomous processors communicating over a communication network and having the following features:

* **No common physical clock**: This is an important assumption because it introduces the element of “distribution” in the system and gives rise to the inherent asynchrony amongst the processors
* **No shared memory:** This is a key feature that requires message-passing for communication. This feature implies the absence of the common physical clock. It may be noted that a distributed system may still provide the abstraction of a common address space via the distributed shared memory abstraction. Several aspects of shared memory multiprocessor systems have also been studied in the distributed computing literature.
* **Geographical separation:** The geographically wider apart that the processors are, the more representative is the system of a distributed system. However, it is not necessary for the processors to be on a wide-area network (WAN). Recently, the network/cluster of workstations (NOW/COW) configuration connecting processors on a LAN is also being increasingly regarded as a small distributed system. This NOW configuration is becoming popular because of the low-cost high-speed off-the-shelf processors now available. The Google search engine is based on the NOW architecture.
* **Autonomy and heterogeneity:** The processors are “loosely coupled” in that they have different speeds and each can be running a different operating system. They are usually not part of a dedicated system, but cooperate with one another by offering services or solving a problem jointly.

**1.3 Issues in distributed computing**

Distributed systems are considerably more complex than monolithic computing environments, and raise a number of challenges around design, operations and maintenance.

**Increased opportunities for failure:**

* The more systems added to a computing environment, the more opportunity there is for failure.
* If a system is not carefully designed and a single node crashes, the entire system can go down.
* Distributed systems are designed to be fault tolerant, that fault tolerance isn’t automatic or fool-proof.

**Synchronization process challenges:**

* Distributed systems work without a global clock
* It requires careful programming to ensure that processes are properly synchronized to avoid transmission delays that result in errors and data corruption
* In a complex system, such as a multiplayer video game, synchronization can be challenging, especially on a public network that carries data traffic

**Imperfect scalability:**

* Doubling the number of nodes in a distributed system does not necessarily double performance
* Architecting an effective distributed system that maximizes scalability is a complex undertaking that needs to consider load balancing, bandwidth management and other issues.

**More complex security:**

* Managing a large number of nodes in a heterogeneous or globally distributed environment creates numerous security challenges
* A single weak link in a file system or larger distributed system network can expose the entire system to attack.

**Increased complexity:**

* Distributed systems are more complex to design, manage and understand than traditional computing environments.

**2.1 Motivation**

The motivation for using a distributed system is some or all of the following requirements:

**1. Inherently distributed computations:** In many applications such as money transfer in banking, or reaching consensus among parties that are geographically distant, the computation is inherently distributed.

**2. Resource sharing:** Resources such as peripherals, complete data sets in databases, special libraries, as well as data (variable/files) cannot be fully replicated at all the sites because it is often neither practical nor cost-effective. Further, they cannot be placed at a single site because access to that site might prove to be a bottleneck. Therefore, such resources are typically distributed across the system. For example, distributed databases such as DB2 partition the data sets across several servers, in addition to replicating them at a few sites for rapid access as well as reliability.

**3. Access to geographically remote data and resources**: In many scenarios, the data cannot be replicated at every site participating in the distributed execution because it may be too large or too sensitive to be replicated. For example, payroll data within a multinational corporation is both too large and too sensitive to be replicated at every branch office/site. It is therefore stored at a central server which can be queried by branch offices. Similarly, special resources such as supercomputers exist only in certain locations, and to access such supercomputers, users need to log in remotely. Advances in the design of resource-constrained mobile devices as well as in the wireless technology with which these devices communicate have given further impetus to the importance of distributed protocols and middleware.

**4. Enhanced reliability:** A distributed system has the inherent potential to provide increased reliability because of the possibility of replicating resources and executions, as well as the reality that geographically distributed resources are not likely to crash/malfunction at the same time under normal circumstances. Reliability entails several aspects:

* availability, i.e., the resource should be accessible at all times;
* integrity, i.e., the value/state of the resource should be correct, in the face of concurrent access from multiple processors, as per the semantics expected by the application;
* fault-tolerance, i.e., the ability to recover from system failures, where such failures may be defined to occur in one of many failure models.

**5. Increased performance/cost ratio**: By resource sharing and accessing geographically remote data and resources, the performance/cost ratio is increased. Although higher throughput has not necessarily been the main objective behind using a distributed system, nevertheless, any task can be partitioned across the various computers in the distributed system. Such a configuration provides a better performance/cost ratio than using special parallel machines. This is particularly true of the NOW configuration.

In addition to meeting the above requirements, a distributed system also offers the following advantages:

**6. Scalability:** As the processors are usually connected by a wide-area network, adding more processors does not pose a direct bottleneck for the communication network.

**7. Modularity and incremental expandability**: Heterogeneous processors may be easily added into the system without affecting the performance, as long as those processors are running the same middleware algorithms. Similarly, existing processors may be easily replaced by other processors.

**2.2 Models and Architectures of Distributed Systems**

There are two Models and Architecture of distributed systems:

1. **Client-server systems:**

* The most traditional and simple type of distributed system, involve a multitude of networked computers that interact with a central server for data storage, processing or other common goal
* The client requests a resource and the server provides that resource
* A server may serve multiple clients at the same time while a client is in contact with only one server
* Both the client and server usually communicate via a computer network and so they are a part of distributed systems.
* Cell phone networks are an advanced type of distributed system that share workloads among handsets, switching systems and internet-based devices

1. **Peer-to-peer networks:**

* Workloads are distributed among hundreds or thousands of computers all running the same software
* The peer to peer systems contains nodes that are equal participants in data sharing
* All the tasks are equally divided between all the nodes
* The nodes interact with each other as required as they share resources
* This is done with the help of a network.

The most common forms of distributed systems in the enterprise today are those that operate over the web. They hand off workloads to dozens of cloud-based virtual server instances that are created as needed, then terminated when the task is complete.