Network Communication Models

In the previous chapter, we saw that data travels across a network in the form of packets. We will now examine the models through which these packets are exchanged between two applications.

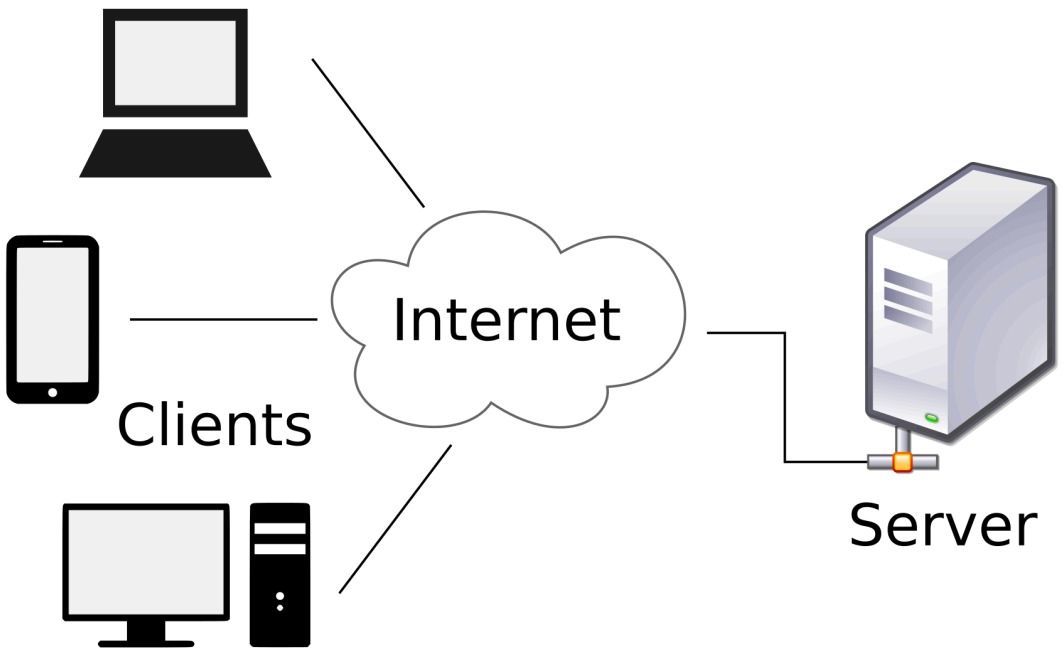
- Client-server model
- Peer-to-peer model

Client-Server Model

The client-server model is a network architecture in which a device or software process, called a client, sends requests to another device or software process, called a server, in order to obtain a service.

The presence of a server allows a number of clients to share its resources, while the server manages access to those resources in order to avoid conflicts.

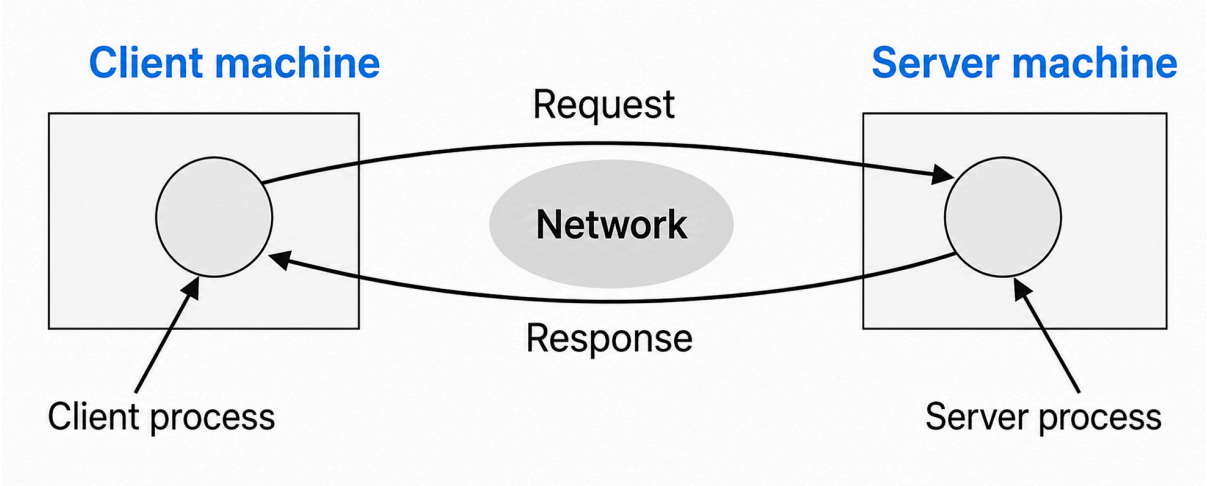
Many services available on the Internet operate according to the client-server model. For example, when we visit a website, the browser sends a request to a web server, which returns the requested page.



More specifically, the client-server model involves two processes, meaning two running programs: one on the client machine and one on the server machine.

Based on these premises, communication follows this logic:

- A client process sends a message across the network to the server process and then waits for a response.
- When the server process receives the request, it processes it and returns a response.



It is important to clarify that a server is not necessarily a physical machine. It can also be a software process that listens on a network port and waits for requests from clients. The same machine can therefore host several servers at the same time, meaning several applications or services running simultaneously, each of which can be accessed by clients. Similarly, a client is not necessarily a physical machine. It can be a software process that sends requests to a server in order to obtain a service. The same machine can run several client processes at the same time, such as a web browser, an email application, an online video game or a messaging application.

Clients and servers communicate through a **communication protocol**, which consists of shared rules that allow them to interact in order to perform a specific task.

Client

The term **client** refers to a computer component or subsystem that accesses the services or resources provided by a server.
A client can be either hardware or software:

- A device connected to a server through a network, from which it requests one or more services using one or more network protocols, is an example of a hardware client.
- An email program, on the other hand, is an example of a software client because the software continuously communicates with the server that receives and sends emails.

One of the most important software clients is the **browser**, a program designed to display web pages retrieved from a server.

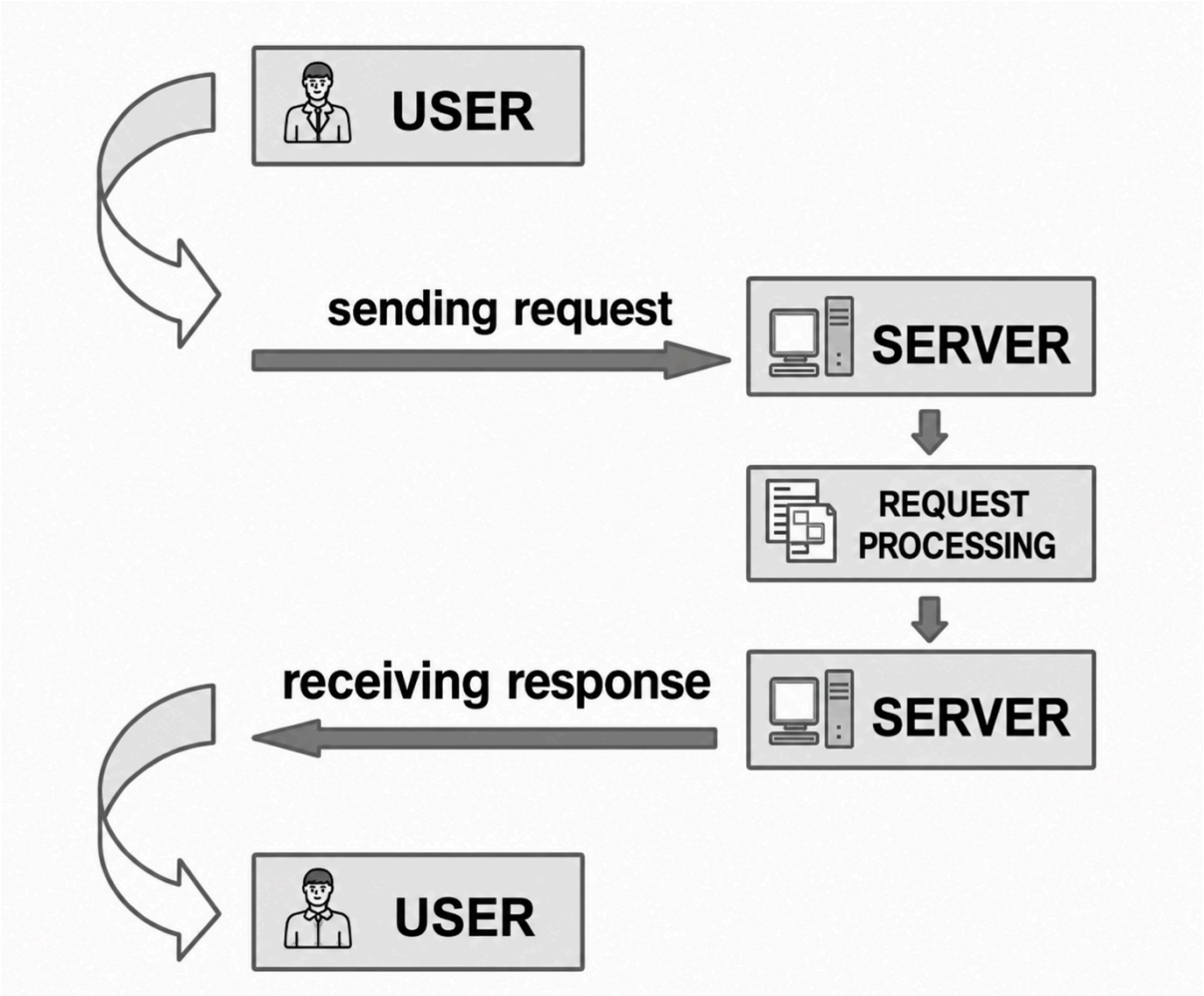
In some systems, the client mainly acts as an interface to the server. In other cases, however, it can be highly complex software capable of processing large amounts of data locally.

Some types of software, such as email systems, are divided into a client component, installed and running on the client computer, and a server component, installed and running on the server.

Example
When Gmail is used through a browser or an application, the user interface acts as the client. The messages, on the other hand, are stored on the service's server systems. The client sends requests to the server, which retrieves the data and returns it so that it can be displayed to the user.

Server

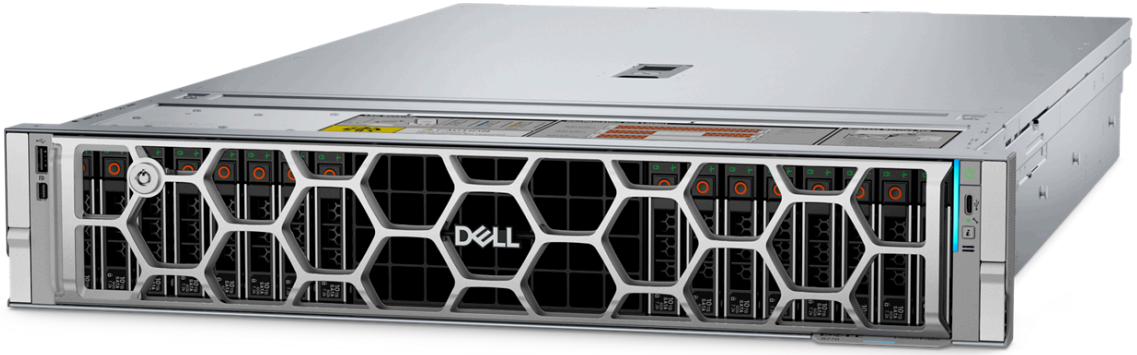
A **server**—from the verb to serve—is a device or software application that provides services to other devices or programs, called clients.



Depending on the context, the term "server" may therefore refer to:

- An ordinary computer used to provide services to other computers, regardless of its hardware characteristics.
- A specialised computer belonging to a product category designed specifically for use as a server, characterised by high reliability, greater performance and additional features.
- A software application that provides services to other software applications.

A server may be responsible for managing user access, allocating and releasing resources, sharing data and protecting information.



Typical Server

Esempio
An email server can be compared to an ordinary post office.
To access their email inbox through a client, users must first be authorised.
Similarly, a person must possess the key to a mailbox located at a post office in order to collect the mail

Tipi di server

In many cases, when we talk about servers, we are referring to software services: programs that perform specific functions and respond to client requests.

The most important services include:

- File server: Allows users to access files stored on a storage device as though they were stored on their own device, making it easier to share information.
- Database server: Manages databases and responds to requests to read, write, modify or delete data.
- FTP server: Provides access over the network to public or authenticated folders.
- Web server: Used to host websites.
- Application server: Runs the logic of a web application, for example by managing functions, users, data and communication between the interface and the database.
- Mail server: Used to manage email.
- Game server: Hosts the resources required to support multiplayer games.

Data center

A data centre is a specialised facility designed to house servers and all the other components connected to them, including storage devices, network equipment, security systems and power infrastructure.

The purpose of a data centre is to store, process and distribute large volumes of data continuously, reliably and securely.

Simply put, data centres are “information factories”: places where data is stored, transmitted and managed 24 hours a day.

A fundamental part of a data centre is the **server farm**, which is an organised group of servers installed in the same physical location.

A server farm is a general term used to describe an organised group of servers installed in the same physical location.

A server farm consists of metal cabinets called racks, each of which contains dozens of servers.

Structure of a Data Center

DATA CENTER

Building | Power (UPS, Generators) | Cooling | Security | Network

SERVER FARM

(A set of racks with many servers)

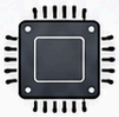
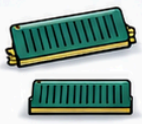
RACK (cabinet)

Contains **20–40 servers**

Components in a Rack:

- Server
- Storage (HDD/SSD)
- Network switch
- Connection cables

SERVER (single machine)

CPU	RAM	Storage	Network card	Other components
				• • •

These servers host websites, databases, business applications, streaming videos and even the virtual worlds of online video games.
In addition to the server farm, a data centre is equipped with:

- Cooling systems, which keep temperatures within acceptable limits because servers generate a great deal of heat.
- Uninterruptible power supplies, or UPS systems, and generators, which guarantee power even during a blackout.
- Very high-speed fibre-optic connections.
- Physical security systems and constant digital monitoring, which prevent unauthorised access and equipment failures.

The entire system is designed to ensure high availability, meaning continuous operation even under critical conditions.
An interruption lasting only a few seconds can result in enormous financial losses or reputational damage for digital service providers.
Without data centres, the Internet as we know it would not exist.
Most of the digital services we use every day, such as email, messaging, video streaming, social networks and cloud platforms, rely on one or more data centres.
Corporate information systems, banks, hospitals, navigation applications and digital public services also rely on this infrastructure.

Rack containing servers

Peer-to-Peer System

The client-server model is the most widely used model on the Internet, but another popular model also exists: **peer-to-peer**, or ***P2P***.

A peer-to-peer network is a network model in which connected devices, called **peers**, can communicate directly with one another without necessarily relying on a central server.

The English word peer means “equal”. This helps us understand the main concept: in a peer-to-peer network, computers are not rigidly divided into clients and servers, but can perform both roles.

A device can therefore:

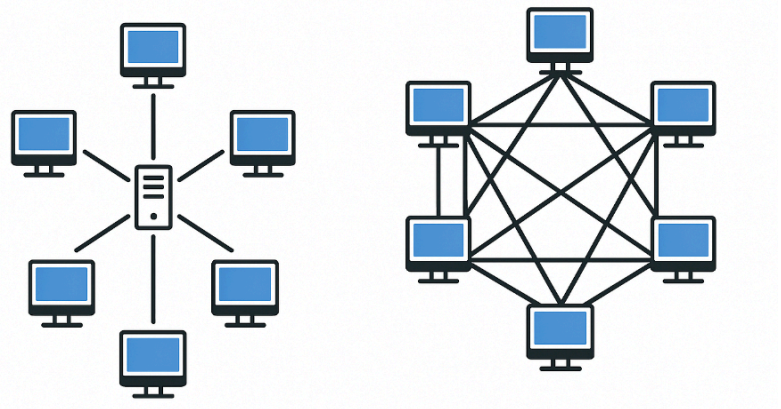
- Request data from another device.
- Send data to other devices.
- Share files, resources or services.
- Contribute to the general operation of the network.

Each peer can act both as a client, when it requests a resource, and as a server, when it provides one.

In the client-server model, communication is organised around one or more main servers. Clients do not always communicate directly with one another, but often communicate through the server.

For example, when we visit a website, our browser is the client and the web server is the system that hosts the requested pages. The client sends a request, and the server responds by sending the required data.

In the peer-to-peer model, on the other hand, devices can exchange data directly.
There is not necessarily a central server controlling all communication.
This is the most important difference: the client-server model is more centralised, while the peer-to-peer model is more distributed.



On the left, a client-server system; on the right, a P2P system.

Each peer can make some of its resources available, such as:

- Files
- Storage space
- Processing power
- Network bandwidth
- Information required for the operation of the system

For this reason, the overall operation of a peer-to-peer network depends on collaboration among many peers.

Example
Imagine a classroom in which every student has a different file stored on their computer.

- In the client-server model, all files would be uploaded to a single central computer, such as the teacher's computer or a school server. Students would download the files from there.
- In the peer-to-peer model, each student could share their file directly with the others. When a student needs a file, they could receive it directly from a classmate's computer.

Not all peer-to-peer networks operate in the same way. In simplified terms, we can distinguish between **pure *P2P* networks** and hybrid ***P2P* networks**.

- In a pure P2P network, there is no central server. All peers have a similar role and collaborate directly with one another. The network is completely distributed.
- In a hybrid P2P network, a central server may help peers find one another, while the actual data exchange takes place directly between the devices. For example, a server might contain a list of available peers without directly hosting all the files being exchanged. In this case, the server does not remove the peer-to-peer nature of the network because the main data is still transferred between peers.

One of the best-known uses of peer-to-peer networks is file sharing.
In a P2P file-sharing system, a file can be divided into many parts.
Each peer can download different parts of the file from different peers while simultaneously sharing the parts it already possesses with other users.
This mechanism is very different from a traditional download from a single server. In the client-server model, when many users download the same file from the same server, that server can become overloaded. In the P2P model, however, the more users participate in the network, the more possible download sources become available.
For this reason, P2P can be highly efficient when distributing large quantities of data.

The peer-to-peer model offers several advantages:

- Load distribution: Because there is not necessarily a single central server, the workload can be divided among many devices.
- Scalability: In some *P2P* systems, when new peers join the network, they do not only increase the number of requests; they also increase the available resources. Each new peer can contribute bandwidth, data or processing capacity.

- Fault tolerance: If a central server stops working, a client-server system may become inaccessible. In a *P2P* network, on the other hand, the failure of a single peer does not necessarily stop the entire network.

However, the peer-to-peer model also has some disadvantages:

- Management and control: In a client-server system, it is easier to control who can access the data, which resources are available and how they are managed. In a P2P network, the nodes are distributed and may join or leave the network at any time.
- Security: Because data may come from different peers, it is important to verify that it has not been altered and does not contain malicious software.
- Resource reliability: If only a small number of peers share a particular file, the file may become difficult to obtain. Furthermore, if many peers disconnect, the quality of the service may decrease.

One of the fundamental questions in a peer-to-peer network is: how does a peer find other peers?
In the client-server model, the client knows the server's address, for example through a URL or an IP address. In the P2P model, however, there may be many peers, and they may change over time and connect from different networks.

To solve this problem, P2P networks can use different discovery mechanisms.

- Some use support servers that help peers obtain a list of other available peers.
- Other networks use more distributed systems. In this case, each node knows only part of the network and can provide information about other nodes to contact. When a new peer joins the network, it begins with a number of previously known nodes. It obtains information about other peers from those nodes and gradually expands its knowledge of the network.

Torrent

One of the best-known examples of peer-to-peer technology is **BitTorrent**.
In everyday language, however, the word "**torrent**" is often used to refer to a download system based on the BitTorrent protocol.

BitTorrent is a communication protocol designed to distribute files among many devices connected to a network.
Unlike a traditional download, in which a client downloads a file from a single server, a torrent system distributes the file among many users.
Every user participating in the exchange is a peer. This means that they can both receive data and send it to others.

In a traditional download, the process is relatively simple: the client sends a request to a server, and the server sends the requested file.
For example, when we download a program from the manufacturer's official website, our computer connects to the company's server and receives the complete file from it.

In a torrent system, however, the file does not necessarily come from a single server. The file is divided into many small parts, called pieces or blocks. Our computer may download some pieces from one peer, other pieces from another peer, and so on.
At the same time, while downloading, we can share the pieces we have already received with other users.
This is the fundamental point: in a torrent system, users who are downloading a file can also contribute to its distribution.

To start a torrent download, a small file with the .torrent extension is often used.
This file does not contain the actual content that we want to download. For example, when we want to download a video, the .torrent file does not contain the complete video.
Instead, the .torrent file contains information that is useful to the torrent program, such as:

- The name of the file to be downloaded.
- The size of the file.
- The way the file is divided into pieces.
- Information used to verify that the downloaded pieces are correct.
- Instructions for finding other peers.

The .torrent file is therefore a kind of “map” that allows the program to understand what it needs to find and how it should reconstruct the final file.

To use a torrent, a dedicated program called a **torrent client** is required.
Examples of torrent clients include:

- qBittorrent
- Transmission
- µTorrent
- BitTorrent Web

The torrent client reads the .torrent file or a link called a **magnet link**, searches for other available peers and begins downloading the pieces of the file.
The client also reassembles the downloaded pieces in the correct order until the complete file has been obtained.

A magnet link is a link containing the essential information required to identify the file to be downloaded. Instead of downloading a .torrent file first, the user can open the magnet link directly with the torrent client.
The client uses this information to search for other peers and retrieve the data required to begin the download.
From the user’s point of view, a magnet link is often more convenient because it is sufficient to click on a link.

Two terms are often used in torrent systems: **seed** and **leech**.

- A seed is a peer that already possesses the complete file and is sharing it with other users.
- A leech, or more precisely a peer that is still downloading, is a user who has not yet downloaded the complete file. Even though the peer does not yet possess the entire file, it can still share the pieces it has already received with others.

The more seeds there are, the easier and faster it is to download the file because more complete copies are available on the network.
When there are only a few seeds, the download may be slow or may stop because some pieces of the file may be difficult to find.

The group of peers sharing the same file is called a **swarm**.
A swarm therefore consists of all the users participating in the distribution of a particular item of content: some possess the complete file, while others possess only part of it.

Torrent systems mainly use two methods to find the peers belonging to the swarm of a particular item of content: **the tracke**r and **the *DHT***.

A tracker is a server that helps peers find one another. It does not necessarily contain the file to be downloaded, but it stores information about the peers participating in the swarm.
When a torrent client wants to download a file, it can contact the tracker to obtain a list of other available peers.

DHT stands for Distributed Hash Table. It is used to find other peers without depending on a single central server.
Instead of asking a tracker which users are sharing a file, the torrent client queries a distributed network made up of many peers.
We can imagine it as a large directory divided among many computers: each peer knows only part of the information, but by collaborating with the others, it becomes possible to find the peers that are sharing a particular torrent.
Through the *DHT*, the client does not query every peer on the network.
Instead, it uses the information contained in the torrent to progressively contact selected nodes in the distributed network until it finds peers sharing that content.