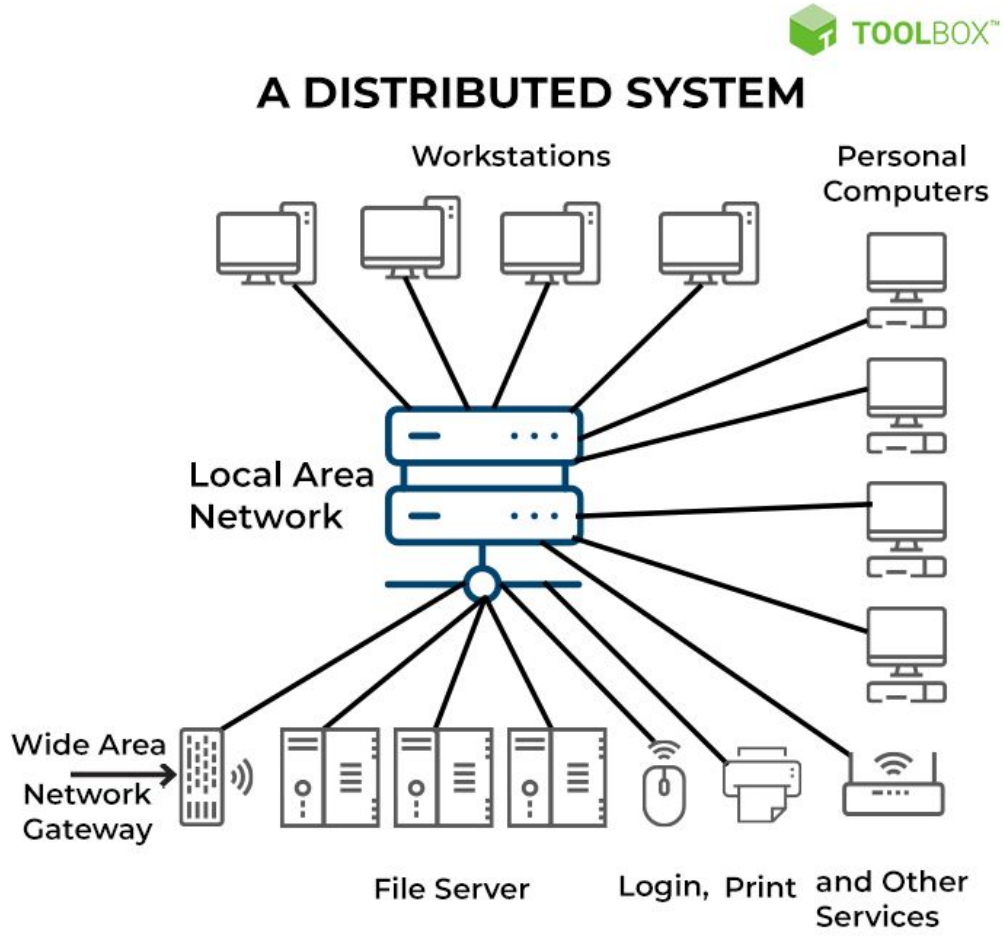
Distributed Computing components

What Is Distributed Computing?

- Distributed computing is a system of software components spread over different computers but running as a single entity.
- A distributed system can be an arrangement of different configurations, such as mainframes, computers, workstations, and minicomputers.

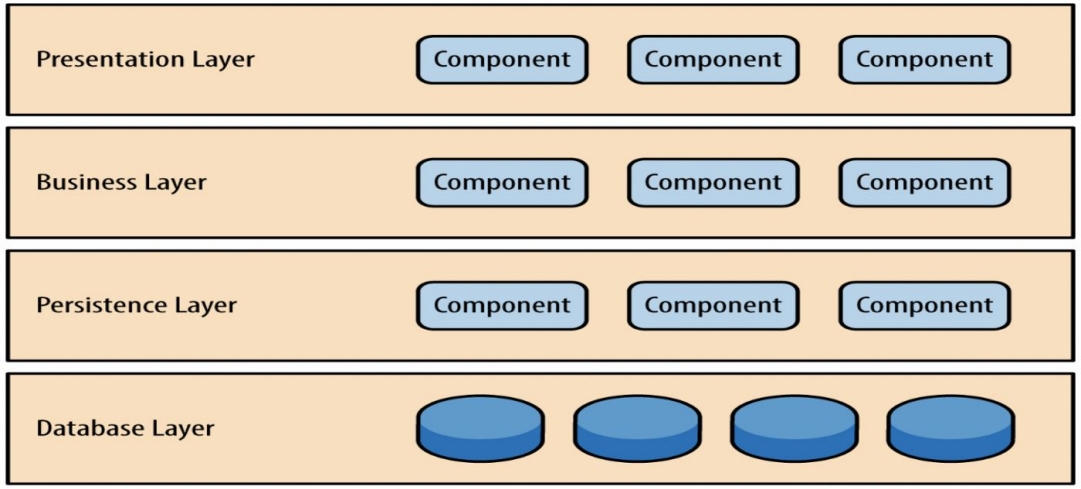


Architecture of Distributed Systems

- 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)
  - System architecture (further divided into client-server architecture and peer-to-peer architecture).

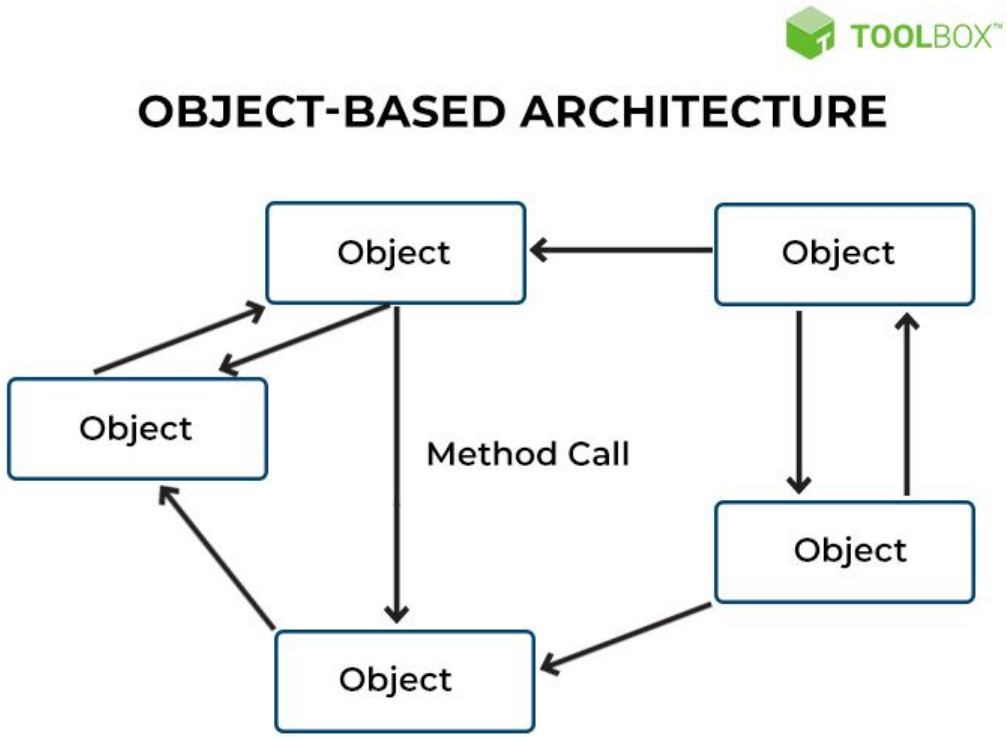
1. Software architecture

- 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:
  - **Layered architecture:** provides a modular approach to software by separating each component. 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 patterns are n-tiered patterns where the components are organized in horizontal layers. This is the traditional method for designing most software and is meant to be self-independent. This means that all the components are interconnected but do not depend on each other.



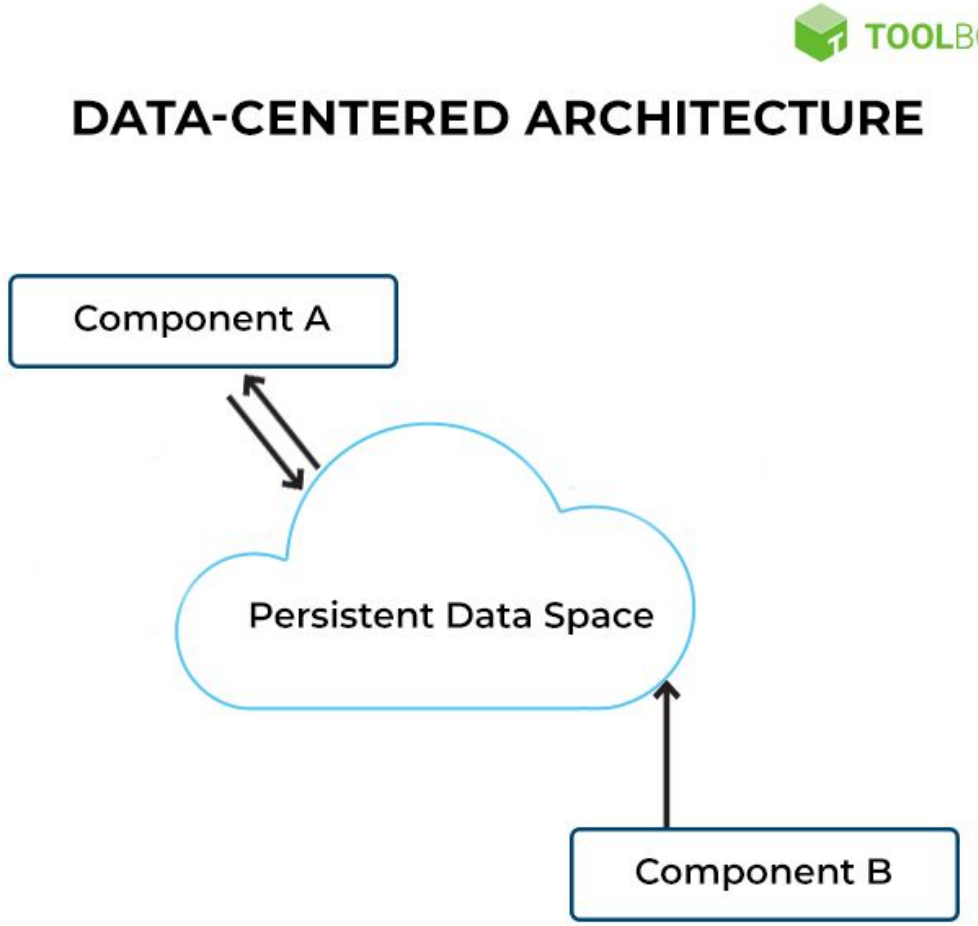
1. Software architecture

- **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.
- 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.



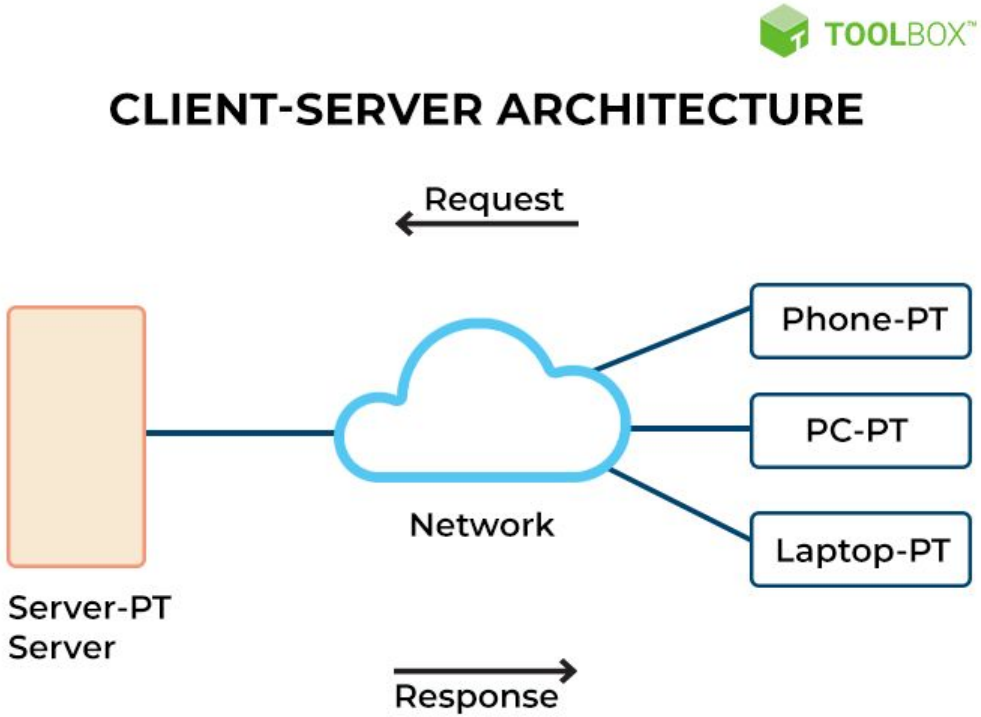
1. Software architecture

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



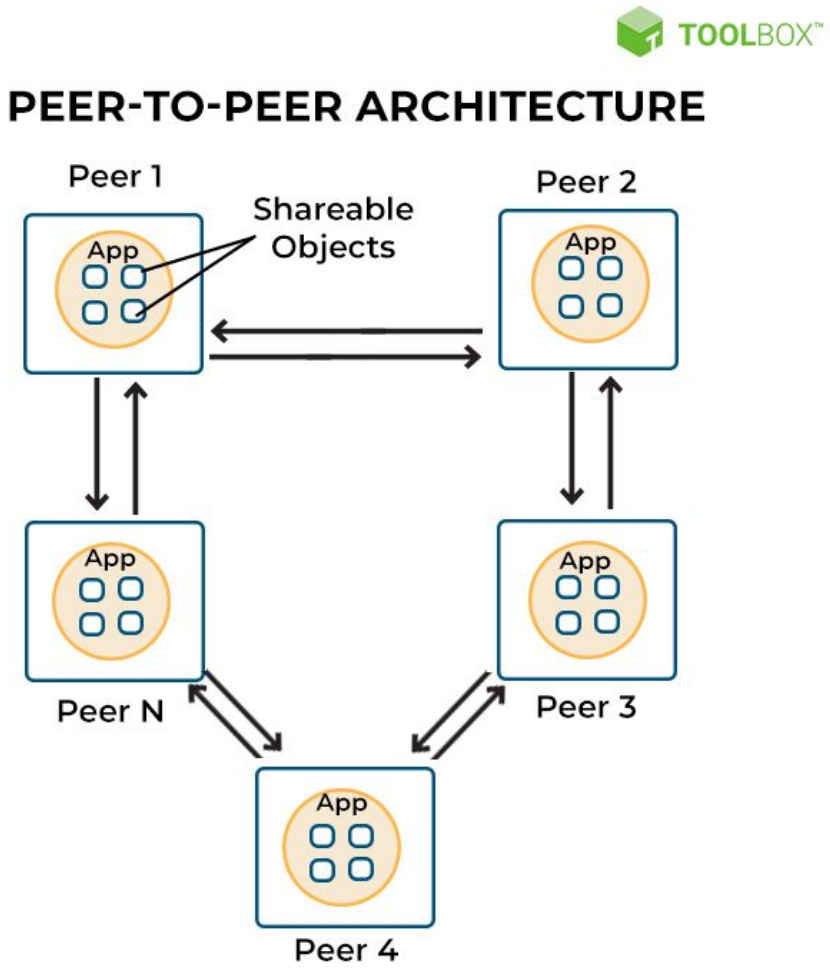
2. System architecture

- 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.
  - **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 has one standard design feature: centralized security.
  - Data such as usernames and passwords are stored in a secure database for any server used 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.



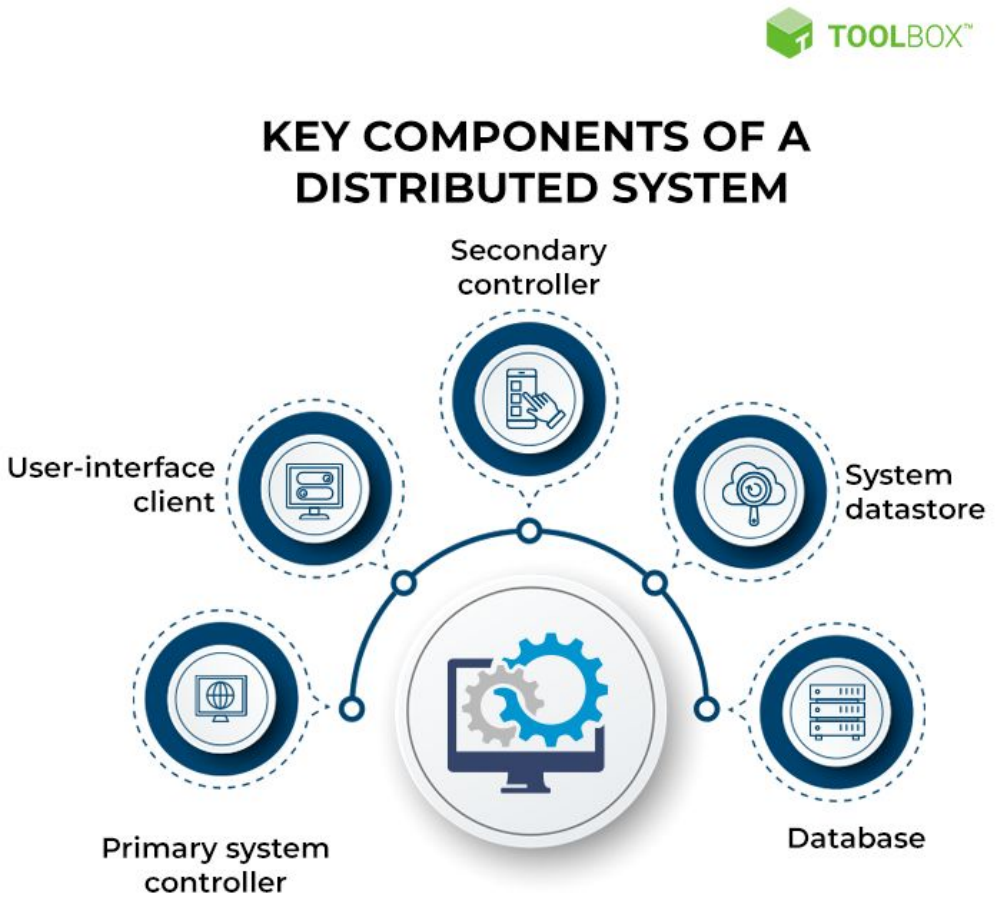
2. System architecture

- **Peer-to-peer 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.
- 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.



Key Components of a Distributed System

- The three basic components of a distributed system include primary system controller, system data store, and database. In a non-clustered environment, optional components consist of user interfaces and secondary controllers.



1. Primary system controller

- The primary system controller is the only controller in a distributed system and keeps track of everything. It's also responsible for controlling the dispatch and management of server requests throughout the system. The executive and mailbox services are installed automatically on the primary system controller. In a non-clustered environment, optional components consist of a user interface and secondary controllers.

2. Secondary controller

- The secondary controller is a process controller or a communications controller. It's responsible for regulating the flow of server processing requests and managing the system's translation load. It also governs communication between the system and VANs or trading partners.

3. User-interface client

- The user interface client is an additional element in the system that provides users with important system information. This is not a part of the clustered environment, and it does not operate on the same machines as the controller. It provides functions that are necessary to monitor and control the system.

4. System datastore

- Each system has only one data store for all shared data. The data store is usually on the disk vault, whether clustered or not. For non-clustered systems, this can be on one machine or distributed across several devices, but all of these computers must have access to this datastore.

5. Database

- In a distributed system, a relational database stores all data. Once the data store locates the data, it shares it among multiple users. Relational databases can be found in all data systems and allow multiple users to use the same information simultaneously.