Distributed Systems:

A system in which hardware or software components located at networked computers communicate and coordinate their actions only by message passing

Cluster:

A type of parallel or distributed processing system, which consists of a collection of interconnected stand-alone computers cooperatively working together as a single, integrated computing resource

Cloud:

a type of parallel and distributed system consisting of a collection of interconnected and ***virtualised computers*** *that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers*

Benefits of building Distributed Systems:

- Functional Separation:
  - Existence of computers with different capabilities and purposes:
    - Clients and Servers
    - Data collection and data processing
- Inherent distribution:
  - Information:
    - Different information is created and maintained by different people (e.g., Web pages)
  - People
    - Computer supported collaborative work (virtual teams, engineering, virtual surgery)
  - Retail store and inventory systems for supermarket chains (e.g., Coles, Woolworths)

Ahmed Diab 🦅❤️

- Power imbalance and load variation:
    - Distribute computational load among different computers.
- Reliability:
    - Long term preservation and data backup (replication) at different locations.
- Economies:
    - Sharing a printer by many users and reduce the cost of ownership.
    - Building a supercomputer out of a network of computers.

Challenges in Distributed Systems:

- Concurrency
- Heterogeneity
- Openness
- Security
- Scalability
- Failure handling
- Transparency

Compare between Parallel and Distributed System:

	Parallel	**Distributed**
Hardware	Identical processors Regular interconnection	Different types of processors Networks
Memory	Shared memory	Distributed memory
Control	Synchronized (global clock)	Synchronized execution of tasks (no global clock)
Main focus or goals	Performance	Information/Resource sharing Reliability/Availability Security ....
Task Homogeneity	Tasks perform similar functions	Heterogeneous, tasks perform different functions

What’s the different between authentication and authorization:

Authentication:

Determines whether users are who they claim to be

Challenges the user to validate credentials (for example, through passwords, answers to security questions, or facial recognition)

Authorization:

Determines what users can and cannot access

Ahmed Diab 🦅❤️

What’s the different between Confidentiality and Integrity?

Confidentiality:

Protection against disclosure to unauthorized individual information

Integrity:

Protection against alteration or corruption

Architectural Model:

- Simplifies and abstracts the functions of individual components
- The placement of the components across a network of computers – define patterns for the distribution of data and workloads
- The interrelationship between the components – i.e., the components function roles and the patterns of communication between them.

Architectural Elements:

1- Objects:

Objects have been introduced to enable and encourage the use of object-oriented approaches in distributed systems

2- Components:

Problem-oriented abstractions for building distributed systems and are also accessed through interfaces

3- Webservices:

i. Web services represent an important paradigm for the development of distributed systems
ii. Integrated into the world wide web

what's the standard for IDL? And What to use?

- interface definition language
- Objects are accessed via interfaces, with an associated it providing a specification of the methods defined on an object

How do those entities communicate in a distributed system?

1- **Inter process communication** refers to the **relatively low-level support for communication** between processes in distributed systems, including message-passing primitives, direct access to the API offered by Internet protocols

2- **Remote invocation** covers a range of techniques based on a two-way exchange between communicating entities in a distributed system and **resulting in the calling of a remote operation, procedure or method**

3- **Indirect communication**

- ***Group communication*****: Group communication is concerned with the delivery of messages to a set of recipients and hence is a multiparty communication paradigm supporting one-to-many communication.**
- ***Publish-subscribe systems***
- **Message queues**

Middleware Examples:

- **<u>Sun RPC (Remote Procedure Call)</u>**
- OMG CORBA (Common Object Request Broker Architecture)
- Microsoft D-COM (Distributed Components Object Model)
- **<u>Sun Java RMI (Remote Method Invocation)</u>**
- Modern Middleware Examples:
    - Manjrasoft Aneka– for Cloud computing
    - IBM WebSphere
    - Microsoft .NET
    - **<u>Sun J2EE</u>**
    - Google AppEngine
    - Microsoft Azure

Socket

- **is an endpoint of a two-way communication link between two programs running on the network**
- **is bound to a port number so that the transport layer can identify the application that data is destined to be sent.**

Client-Server Network

A centralized server is used to store the data because its management is centralized. In Client-Server Network, the Server responds to the services which are requested by the Client.

Peer-to-Peer Network

This model does not differentiate between the clients and the servers, In this model every node is itself client and server. In Peer-to-Peer Network, Every node can do both requests and responses for the services

Client

A computer running a program that makes a request for services.

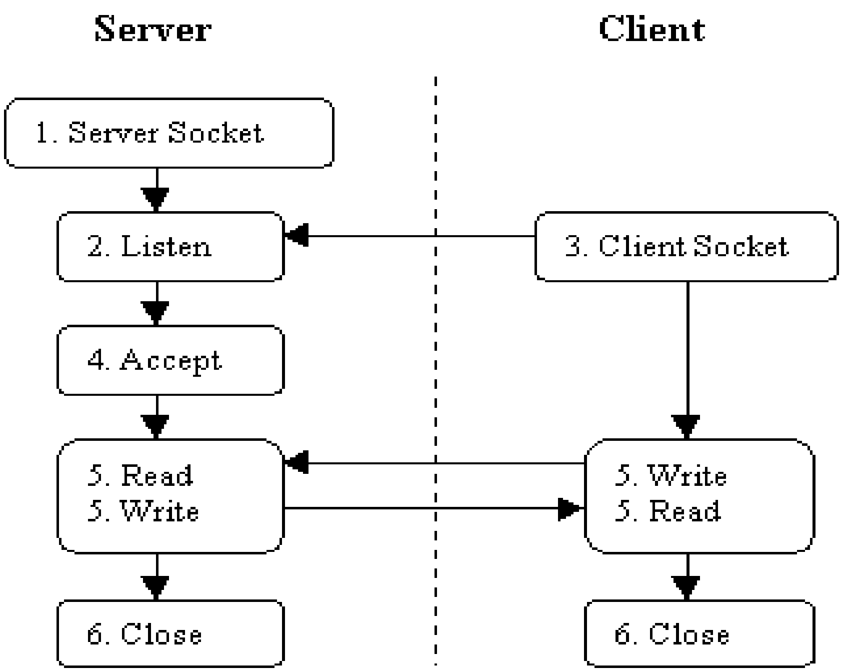
Server

A computer running a program that offers requested services from one or more clients

What's the difference between TCP and UDP?

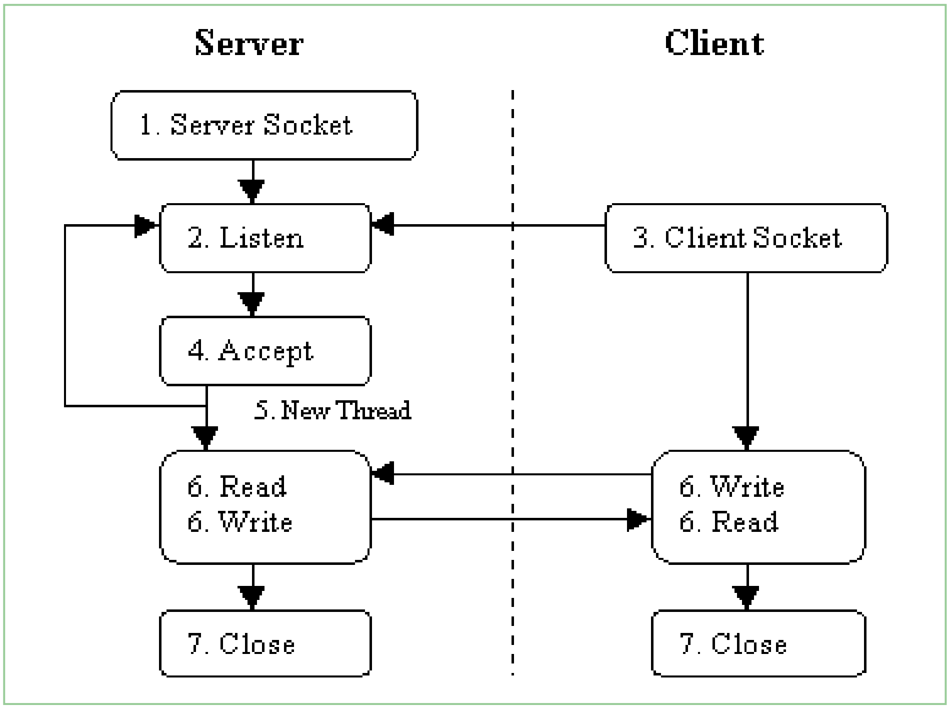
- TCP is a ***reliable*** protocol, UDP is not
- TCP is ***connection-oriented***, UDP is ***connectionless***
- TCP incurs overheads, UDP incurs fewer overheads
- **UDP is simple and efficient**

TCP Socket Communication



Multi-threaded Servers

A server should be able to serve multiple clients simultaneously



Threads States:

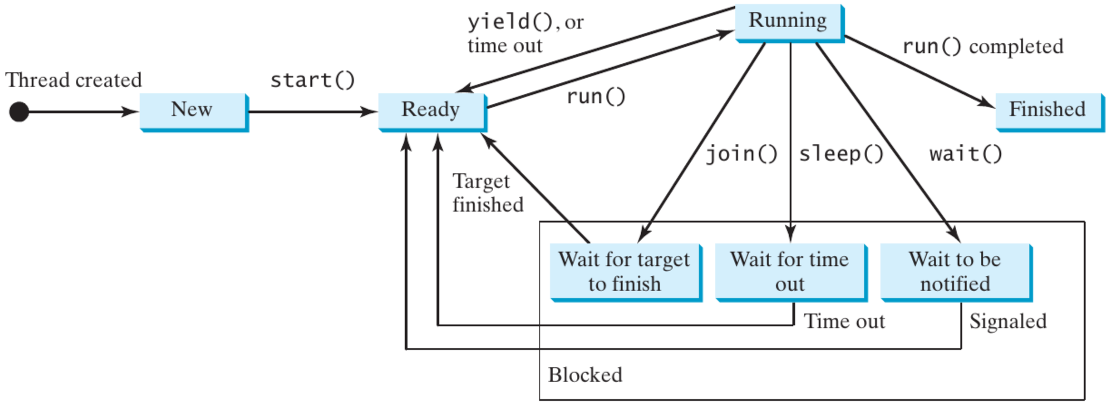


FIGURE 30.25 A thread can be in one of five states: New, Ready, Running, Blocked, or Finished.

Thin-Client model:

All application processing and data management by server only.

Thick-Client model :

Server only responsible for data management. The client machine implements application logic and interactions with user.

Three-tier client-server:

There is a layer between client and server that may provide data and/or application processing.

What’s the difference between **Thin-Client and Thick-Client?**

Thin-Client:

Pros:

- **Access to legacy systems**
- **System management and administration**

- **from admin perspective: system maintenance, security**
- **from user perspective: not hassle with administrative aspects or constant upgrades**
- **More security**
- **Green IT (power saving --> cost saving)**

Cons:

- **Heavy processing load on both server and network (bottleneck)**
- **Less client-perceived performance (in highly interactive graphical activities such as CAD and image processing)**
- **Need to be always connected**

Thick-Client:

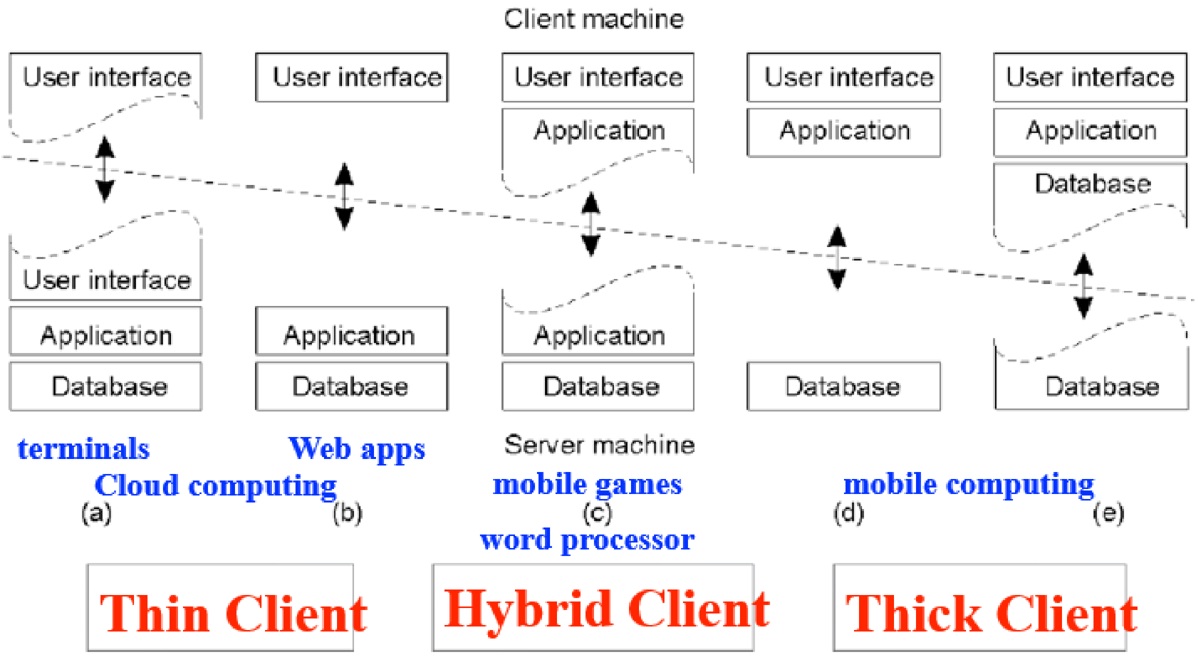
Pros:

- **Better client-perceived performance**
- **(Partly) available offline**
- **Distributed computing (no single point of failures)**
- **Devices are becoming ever faster and cheaper**

Cons:

- **System management and related costs**
- **Having more functionality on the client makes client-side software more prone to errors and more dependent on the client’s underlying platform**

Ahmed Diab 🦅❤️


Client machine
User interface
User interface
User interface
User interface
User interface
Application
Application
Application
Database
User interface
Application
Application
Application
Database
Database
Database
Database
Database
terminals
Web apps
Server machine
Cloud computing
mobile games
mobile computing
(a)
(b)
(c)
(d)
(e)
word processor
Thin Client
Hybrid Client
Thick Client