

Distributed system

Q1) What is a Distributed system? - Explain in details

"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable." by Leslie Lamport.

* A distributed system mainly a system consists of many machine working together.

"A set of nodes, connected by a network, which appear to its users as a single coherent system."

→ we focus on concept, models and many study

* It is important and useful

→ Societal importance

* Internet

* WWW

* Small devices (mobiles, sensors)

Technical importance

* Improve scalability Example — web server use by single server

* Improve reliability — multiple machine, cloud computing

* Inherent distribution. —

Increasing Reliability

* PASS developed by IBM in 1981, used in a Space shuttle.

↳ Could have been done on one node.

↳ But 4 separate nodes used for fault-tolerance.

* It is very challenging, * voting on outcome.

Communication between distributed objects by means of remote method invocation (RMI).

Objects that can receive remote method invocations are called remote objects and they implement a remote interface.

Due to the possibility of independent failure of invoker and invoked objects, RMIs have different semantics from local calls.

They can be made to look very similar to local invocations, but total transparency is not necessarily desirable.

- The code for marshalling and unmarshalling arguments and sending request and reply messages can be generated automatically by an interface compiler from the definition of the remote interface.

Remote procedure call is to RMI as procedure call is to object invocation.

This unit or chapter concerned with programming models for DS application.

→ those ~~progr~~ applications that are composed of cooperating

- programs running in several different processes. Such programs need to be able to invoke operations in other processes, often running in diff. computers. To achieve this, some programming models have been extended to apply to DS programs.

* The earliest best known of these was the extension of the conventional procedure call model to remote procedure call model, — which allows client programs to call procedures in server programs running in separate processes & generally in diff computers from the client.

Q2). Explain client/server model of distributed system.

The Client/Server Model :- Another way of describing process interaction is by how processes view each other.

* one popular approach is the client/server model.

⇒ The client/server model is a programming paradigm that represents the interactions between processes and structures of the system.

Overview :- Any process in a system either provides services for other ~~server~~ or request services from them.

Processes that request services are clients, and those that provide services are servers.

In any interaction a process is either a client or a server.

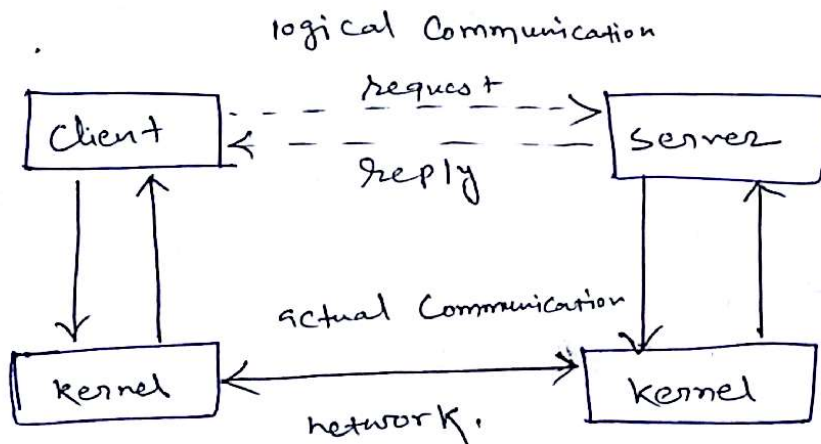


Fig: the concept of the client/server model for process interactions.

A client and a server interact through a sequence of request and responses. A client request a service from the server and blocks itself. The server receives the request, performs the operation, and returns a reply message to the client. The client then resumes its execution.

The only underlying assumption is a synchronous request/reply exchange of info.

⇒ logically, the clients communicate directly with the servers.

Graph models for process Representation ! How (process & thread) they are put together. (4)

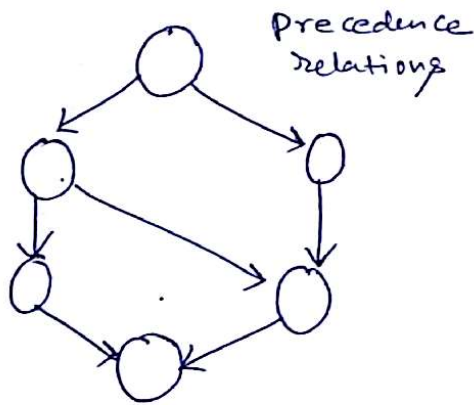
together.

⇒ processes are related by their need for synchronization and/or communication.

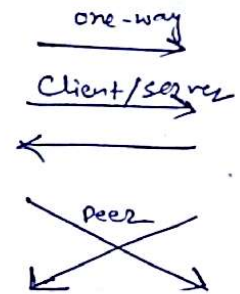
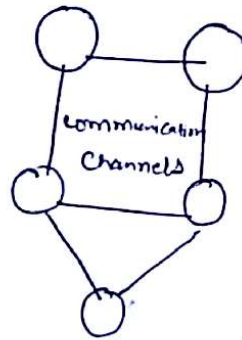
⇒ synchronization means that the execution of some processes must be Serialized in a certain order.

⇒ A special example of synchronization is the precedence relationship between two processes when execution of one process must precede the other.

NOTE - In many cases, the precedence relationship or the ordering of processes is not always necessary for communicating processes as long as messages can be properly exchanged.



Synchronous process graph.



Asynchronous process graph and communication scenarios

Fig: 2.

Fig 2- uses graph models to illustrate these two different views of process interactions.

- The synchronous process graph in the Direct Acyclic Graph (DAG) model shows explicit precedence relationships and a partial ordering of a set of processes.
- The undirected edges in the asynchronous process graph indicate the existence of communication paths, and thus some dependency between processes.

Time services :-

Q3). Difference between physical clock and logical clock.

Physical clocks :- In every computer system, physical clocks are used to synchronize and schedule hardware activities.

On the software side, it is necessary to simulate real time or measure an interval of time.

Software timer based on hardware clock interrupts keep track of software activities.

⇒ In a distributed system, each clock runs at its own pace, and there is a nonnegligible communication delay in reporting clock time.

→ Timing information cannot be propagated and received instantaneously by all, an absolute global physical ~~clock~~ time is theoretically impossible to obtain.

So we have to settle for a close approximation of the global time.

✓ UTC :- universal coordinated time.

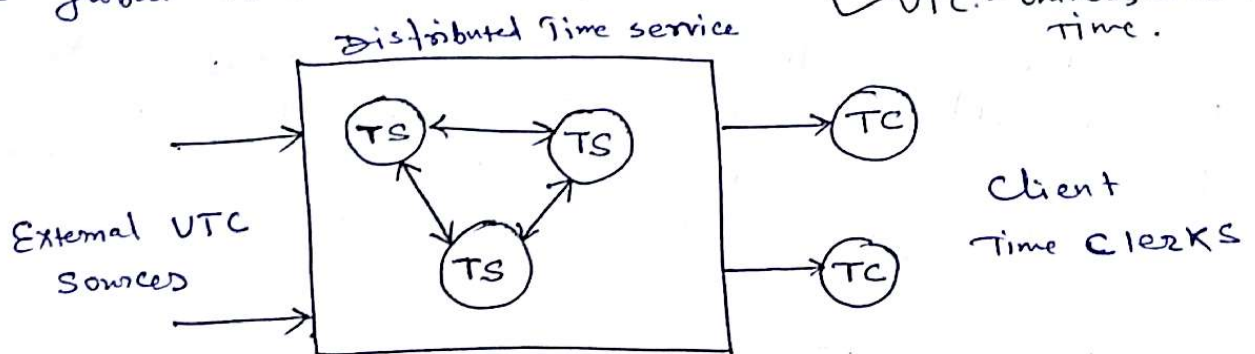


Fig: A distributed time service architecture.

From Fig. A time service architecture that is similar to the distributed time service (DTS) in DCB.

A time clerk (TC) on each client machine may request time service by contacting one or more time servers (TS). Time servers are those that maintain up-to-date clock information and may have access to a universal real time source. Time servers may exchange timing info so that their time services can be more consistent with that of their clients.

Practical Issues :- implementing the time service.

1. The delay in reporting time must be compensated.
2. The discrepancy among time sources must be correct.
(difference)

Compensating Delay:- Fig. shows three types of time access:

- * time server to universal time sources.
- * client to time server.
- * time server to time server.

NOTE:- many standard Universal Coordinated Time (UTC) sources are available for computers and other time critical applications.

* The National Institute of Standards and Technology (NIST) supports several UTC access methods at accuracy near 1 millisecond.

* The Automated Computer Time service (ACTS) provides modem service to NIST time through telephone lines.

* The message delay time to compensate the UTC can be calculated accurately if the distance to the radio station and the signal propagation speed are known.

Calibrating Discrepancy:-

Difference among time servers can be reduced by requiring them to adjust their UTCs cooperatively.

Time servers can periodically exchange UTCs with each other using either the push or pull model.

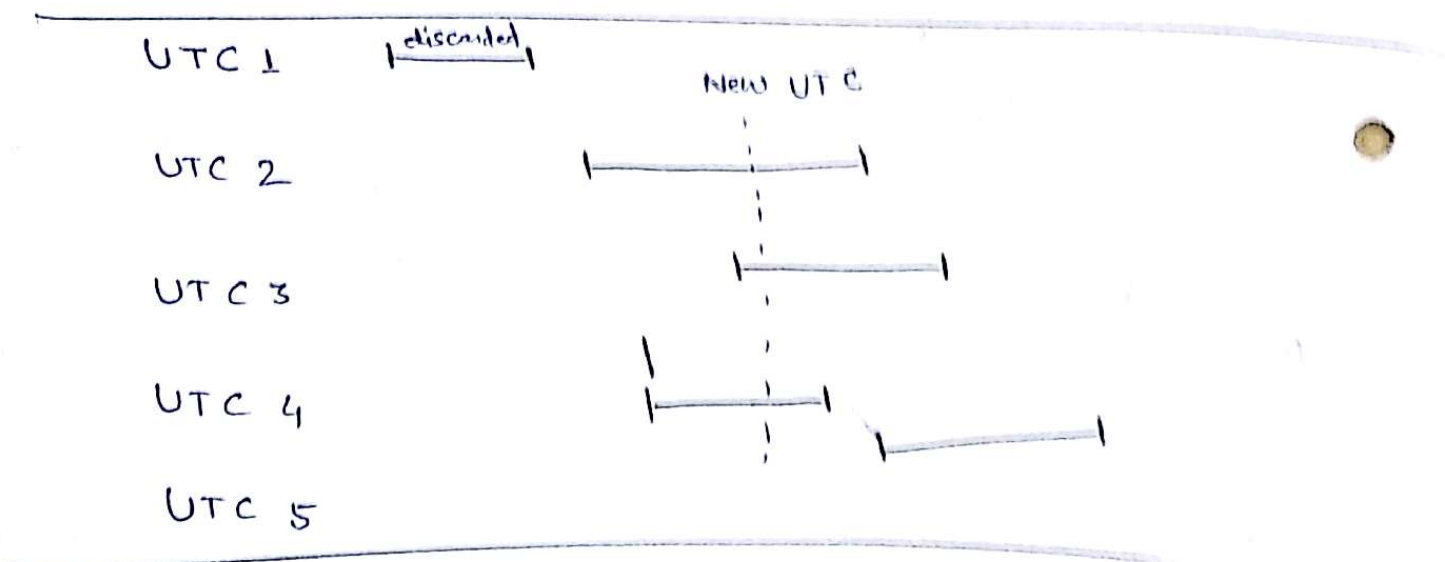


Fig: - Averaging UTC interval.

NOTE:- The physical clock is important for distributed systems development because numerous software protocols rely on a time-out for handling exceptions.

Q.4). Explain Lamport's Logical Clocks.

It is a fundamental concept for ordering processes and events in distributed systems.

Each process P_i in the system maintains a logical clock C_i .

Lamport defines the algebraic notation $\rightarrow$ as the happens-before relation to synchronize the logical clocks between two events.

$[a \rightarrow b]$ means event a precedes event b ,

within a process, if event a occurs before b , logical clocks $C(a)$ and $C(b)$ are assigned such that

$$[C(a) < C(b)].$$

→ The logical clock in a process is always incremented by an arbitrary positive number when events in the process progress.

⇒ Processes interact with each other through a pair of send and receive operations. Sends and receives are considered to be events.

A corresponding send and receive pair from process P_i to process P_j must have the property that $C_i(\text{send}) < C_j(\text{receive})$. Since a receive event cannot be completed until a corresponding send is done — $d=1$

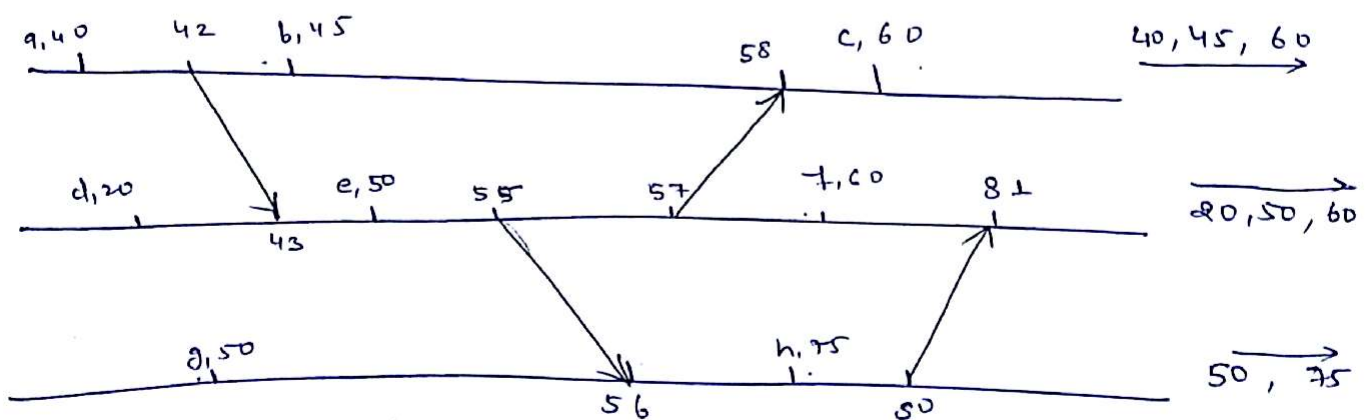


Fig: - The happens-before relation and assignments of logical clock time.

Logical Clocks :-

① ②

A physical clock is a close approximation to the global real time.

The time interval measurement is useful and can be derived directly from physical clocks.

In general we can use physical clocks to tell whether an event happens before another unless they occur very closely.

⇒ If the uncertainty of UTC is high or UTC intervals of events overlap with each other, it is not possible to determine the ordering of events.

For many applications, events need not be scheduled

or synchronized with respect to the real-time clock.

It is only the ordering of event execution that is of concern.

In such cases, logical clocks can be used to indicate the ordering infoⁿ for events.

→

Rule.

1.

2.