Date 16/02
Page 19

TUTORIAL - 6

1119CS012

1.) Explain the three-phase commit protocol.

1.)

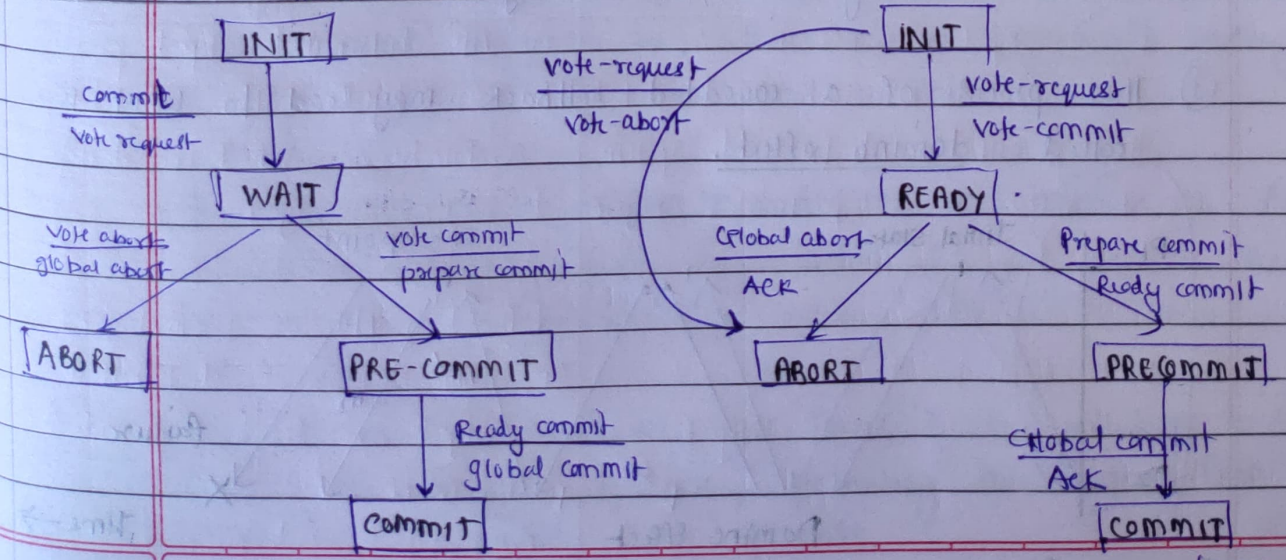
commit (2PC).

① Three Phase commit (3PC) Protocol is an extension of two-phase protocol that avoids blocking problem under certain assumptions. In particular, it is assumed that no network partition occurs and not more than 'k' sites fail, where we assume 'k' ← pre-determined.

② With the mentioned assumptions, protocol avoids blocking by introducing an extra third phase where multiple sites are involved in the decision to commit.

③ Instead of directly noting the commit decision in its persistent storage, the co-ordinator first ensures that at least 'k' other sites know that it intended to commit transaction.

④ In a situation, where co-ordinator fails, a new co-ordinator is selected. If the co-ordinator has decided to commit, at least one of other 'k' sites that it informed will be up and will ensure that commit is done. The new co-ordinator restarts third phase if any of rest sites knew that old co-ordinator intended to commit or else new co-ordinator aborts the transaction.



(a) The final state machine for Co-ordinator

(b) The finate state machine for Participant

2.> Explain your understanding for the following:

a) Orphan Message

① In Client / server Communication, if Client crashes after request, an "old" reply arrives, such as unwanted computation / reply is known as orphan message.

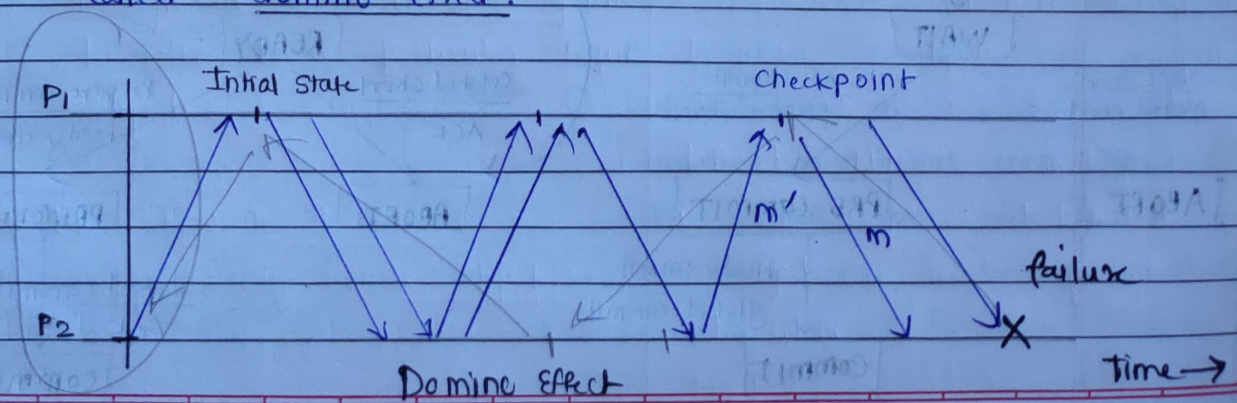
Example: A situation arises that 'A' has received message 'm' from B, but B has no record of sending it, this corresponds to an inconsistent state. "m" is referred an "Orphan Message"

b) Domino Effect

① The distributed nature of checkpointing (in which each process simply record its local state from time to time in an un-co-ordinated fashion) make it difficult to find a recovery line.

② To discover a recovery line requires that each process is rolled back to its most recently rolled state. If these local states jointly do not form a distributed snapshot, further rolling back is necessary.

③ This process of a cascaded rollback may lead to what is called domino effect.



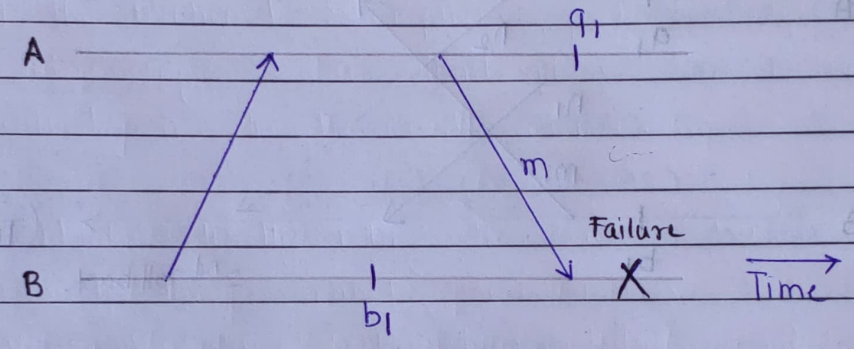
recovery line

c) Lost messages

① Suppose that check points a_1 & b_1 are chosen as recovery point for Process A & B. In this case message 'm' is recorded in a_1, while not in b_1.

② If B fails after message 'm', then system is restored to state $\{a_1, b_1\}$, where message "m" is lost message.

③ This condition can also arise, if 'm' is lost in communication channel.



message 'm' is lost message, due to Roll back recovery.

d) Problem of Live Locks.

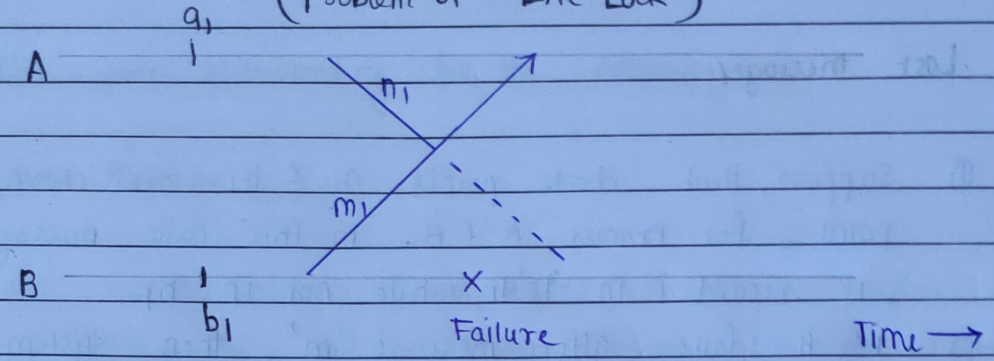
① Livelock is a situation where a single Failure can cause an Infinite number of rollbacks.

② Process B fails before receiving A, sent by A. When B is rolled back to b_1, there is no record of sending m_1. ∴ A is roll-backed to a_1.

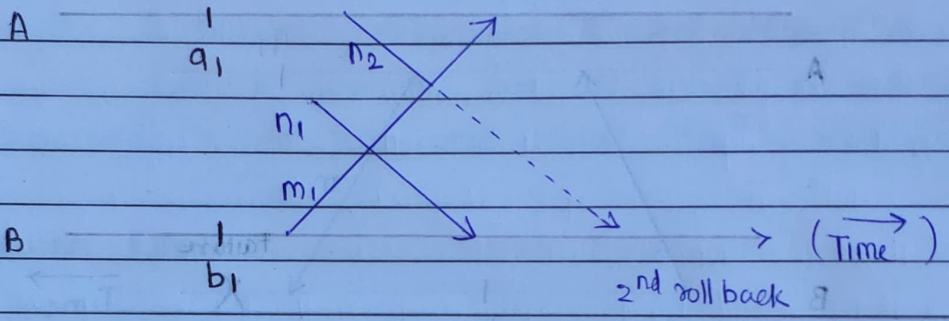
③ When Process B recover it sends m_2 and receives n_2. Process A after resuming from a_1, sends n_2, and receives m_2.
∵ A is ~~roll backed~~ rolled back, there is no record of sending n_1.
∴ B has to roll back. This forces A to roll back to a_1.
This System repeats infinitely, preventing the System from making any progress.

U19CS012

(Problem of Live Lock)



After First recovery



3.> What is difference between by remote objects and distributed objects.

3.> Remote Objects	Distributed Objects
① Object that is hosted by single server, but whose method can be invoked by remote client.	① A distributed object is one whose state may by physically distrib across different servers.
② Implements one or more interfaces that declare remote method of object	② Can be used like regular object, but from anywhere on network.
③ Remote object reference is an identifier that can be used globally through out a distributed system to refer to a particular unique object.	③ All distributed object protocols are built on the same basic architecture which is designed to make an object on one computer looks like its residing on different comp. [Based on Network comm. layer]
④ Most object based systems, support only remote object.	④ An example of distributed object are those provided by Globe.

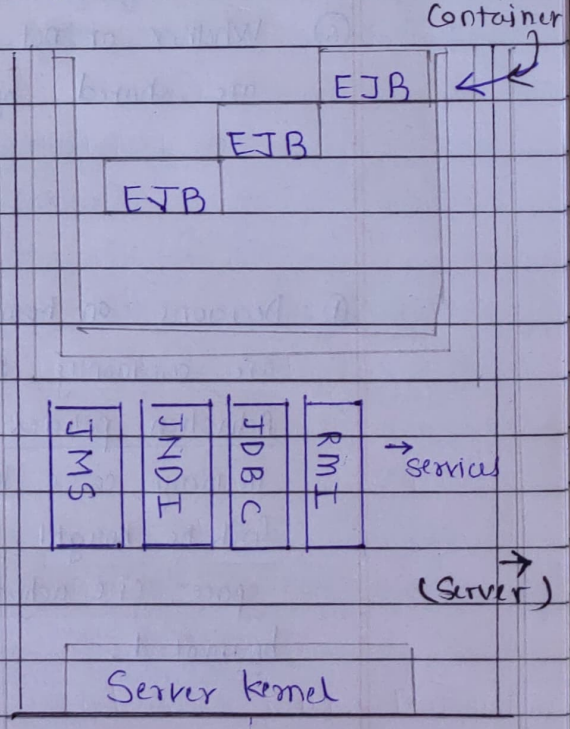
4.) Explain Enterprise Java beans.

4.) ① An Enterprise Java beans is essentially a JAVA object that is hosted by a special server offering different ways for remote clients to invoke that object.

② Crucial is that this server provides the support to seperate application functionality from systems-oriented functionality. The latter includes functions for looking up objects, storing objects, letting objects to be part of a transaction and so on.

③ Typical services (1) Remote method Invocation (RMI)
(2) Database access (JDBC)
(3) naming (JNDI)
(4) Messaging (JMS)

Making use of these services is more or less automated.



④ Stateless session beans are transient objects that are invoked once, does its work, after which it discards any information it maintains to perform the service it offered to a client.

⑤ Stateful session beans maitain client state related

⑥ Entity beans can be considered to be long lived persistent object. Such entity beans will generally be stored in a database and likewise will also be part of distributed transactions.

⑦ Message driven beans used to program objects that should react to incoming messages. They cannot be invoked directly by client but rather fit into publish-subscribe way of communication.

5.> Explain in detail about object server and object adapters. 11

5.>

OBJECT SERVER

① A key role in object based distributed system is played by object servers, that is, the server designed to host distributed objects.

② The important difference between a general object server and other servers is that an object server by itself does not provide specific service.

③ Specific service are implemented by the objects that reside in server.

④ An object server thus acts as a place where object lives.

⑤ Object consist of 2 parts (1) Data representing its state
(2) Code for executing its method

⑥ Whether or not these parts are seperated, or whether method implementation are shared by multiple objects, depend on the object server.

OBJECT ADAPTERS

① Decisions on how to invoke an object are commonly referred to as Activation policies, to emphasize that in many cases the object itself must first be brought into the server's address space (i.e. activated) before it can actually be invoked.

② What is needed then is a mechanism to group objects per policy such a mechanism is sometimes called an object adapter, or alternatively an object wrapper. An object adapter can best be thought as software implementing specific activity policy.

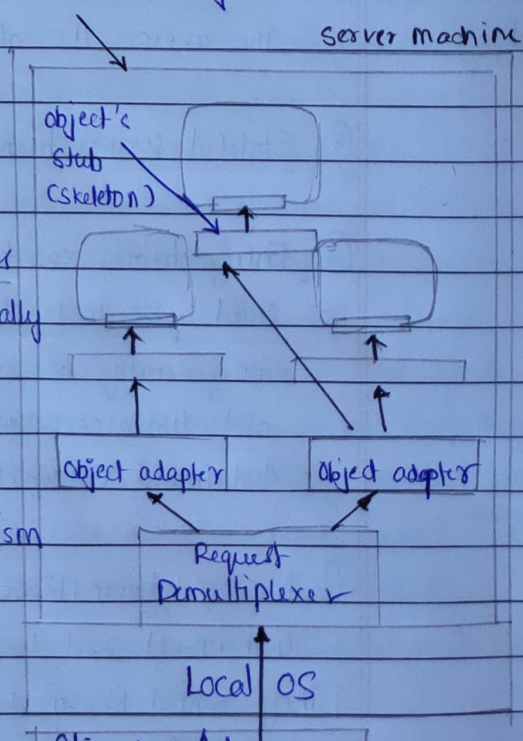
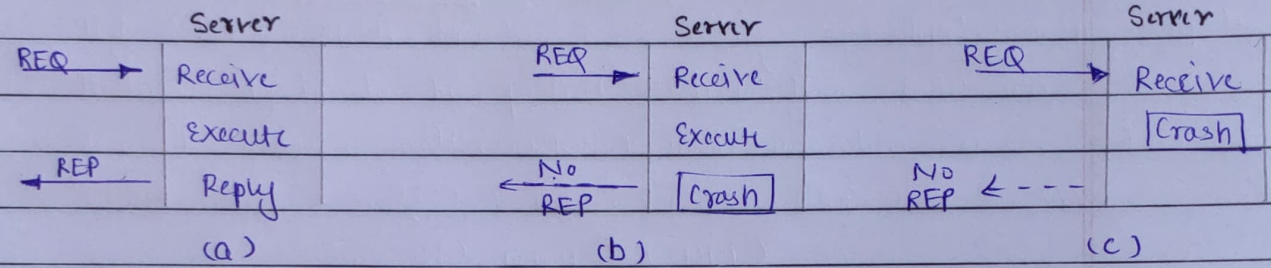


Fig Object Adapters

6.) Write the different problems and different solution with respect to:

a) The server crashes.

PROBLEMS (a) The normal case
(b) Crash after service execution
(c) Crash before service execution



SOLUTIONS ① At least once semantics : Keep trying until a reply is received. Guarantee is given that the RPC occurred at least once but (also) possibly more that once.

② At most once semantics : give up immediately and reports back failure. Guarantee is given that the RPC occurred at most once, but possibly not at all.

③ No Semantics : When a server crashes, client gets no indication. Nothing is guaranteed, and client and server take their chances. Easy to implement.

b) Lost reply messages [PROBLEM - Find out why was there no reply? Server dead, slow or reply went missing?

(Solutions) ① It can be dealt easily by using Timeouts.

② If no ACK arrives in time, the message is resent.

③ Server needs to be able to deal with possibility of duplicate requests.