Name- Mohd. Manzar Iqbal
Scholar No. 19/2007
Branch- MCA - IV Sem

04/05/21

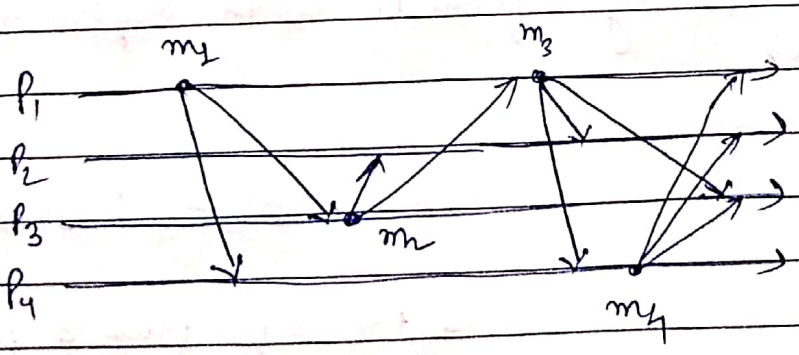
Date:
Page:

Mid-term Exam
MCA-624 Distributive System

(Answer-2) FIFO stands for "first in, first out". It means that if item A is put onto a queue before item B, then item A will come out of the queue before item B.

Casual ordering is more general and is most useful when applied to distributed systems. Casual ordering means if item A reaches a single computer before item B, then item A happens before item B.

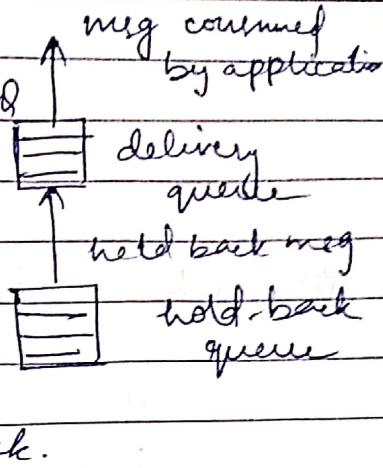
FIFO ordering :-



- with FIFO ordering, messages from a particular process P_i must be received at all other processes P_j in the order they were sent.
  - e.g in the above, everyone must see m_1 before m_3
  - (ordering of m_2 and m_1 is not constrained)
- Hence receivers may need to buffer messages to ensure order.

Implementing FIFO ordering:

```
receive CM from Pi ) {
    S = seqNo(M);
    if (S == (Sji + 1)) {
        S = flush(hbq);
        Sji = S;
    } else holdback(M);
}
```



- each process P_i maintains a message sequence number (SeqNo) S_i.
- Every message sent by P_i includes S_i, incremented after each send. - not including retransmissions.
- P_j maintains S_{ji} the SeqNo. of last delivered message from P_i
- if receive message from P_i with seqNo $\neq (S_{ji}+1)$ hold back

FIFO	Casual Ordering
• Messages from a particular process P_i must be received at all other process P_j in the order they were sent	• Message from a particular process P_i must be received at any order, but delayed to read in order.
• Receiver may need to buffer messages to ensure order	• Receivers may need to delay messages to ensure order.

Date:
Page:

• uses sequence number to create order	• Uses vector clock to create order

(4) Distributed Systems: Distributed systems are basically a group of computers working together as to appear as a single computer to the end user. These machines have a shared state, operate concurrently and can fail independently without effecting the whole system's uptime.

Characteristics of distributed System:

→ Resource Sharing: The resources in a distributin system can be accessed or remotely accessed across multiple computers in the system. The resource can be - hardware (.disks and printers) software and data. Resources are managed by a software module called resource manager.

→ Openers: It is concerned with extensions and improvement of distributed systems. A component can be opener by published specification and interface

→ Concurrency: It refer to the property of a system by which multiple activities are executed at same time, and the activities may perform some kind of interaction.

Date:
Page:

Scalability : It is concerned about the growth of the system as the no. of users of system increases.

Fault tolerance :- The system must be designed in such a way that it is available all the time even after something has failed.

Transparency :- The system should provide transparency of acces location, concurrency, replication, failure, migration, performance scaling, etc.

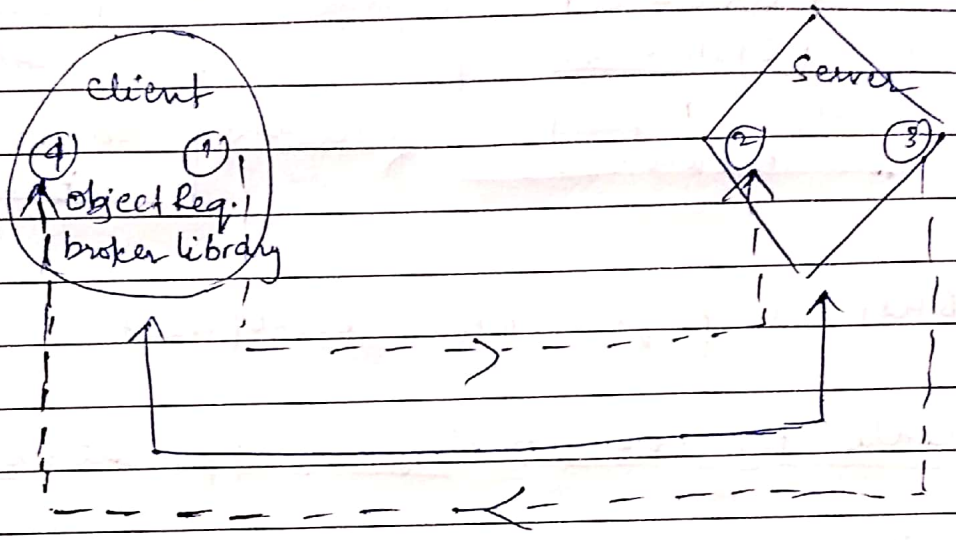
Example of Distributive System

→ Distributive UNIX
→ Wide Area Network application like
- email
- bulletin board system
- WWW
- teleconferencing

(Ans-3) **CORBA** or common Object Request Broker Architecture, is a standard architecture for distributed object systems. It allows a distributed, heterogenous collection of objects to interoperate.

Date:
Page:

- The common Object Request Broker Architecture (CORBA) is a standard developed by the Object management Group (OMG) to provide inter-operability among distributed objects

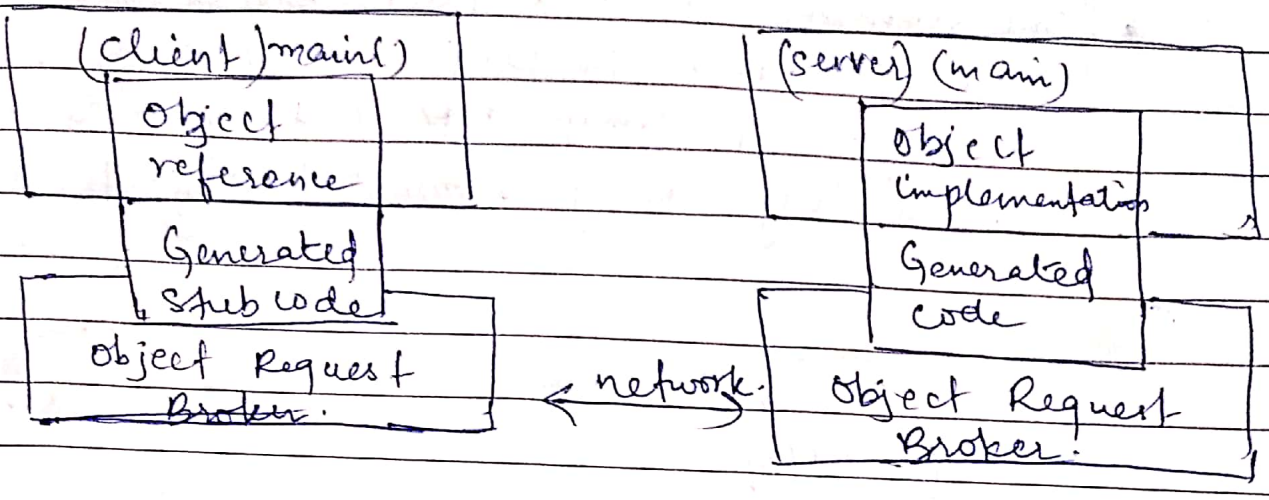


(1) Client sends request to server
(2) Server receives a request from Client
(3) Server sends a reply to the client
(4) Client gets reply from the server

- The CORBA interface Definition Lang. allows the development of language and location-independent interfaces to distributed objects.

- Data communication from client to server is accomplished through a well-defined object oriented interface. The object Request broker determines the location of the target object, sends a request to the object and returns response back to caller.

Date:
Page:



* Basic steps for CORBA development :-

(1) Create the IDL to define Application interface

The IDL provides the operating system and programming language-independent interfaces to all the service and component that are linked to the ORB.

(2) Translate the IDL

An IDL translator typically generates two cooperative parts of the client and server implementation, stub code and skeleton code.

(3) Compile the interface files.

Once the IDL is translated into the appropriate language, these interface files are compiled and prepared for the object implementation.

Date:
Page:

(4) Complete the implementation

If the implementation class are incomplete, the spec and header files and complete bodies and definition need to be modified before passing through to be compiled.

(5) Compile the implementation

Once the implementation class is complete, the client interface are ready to be used in the client application and can be implemented into the client process.

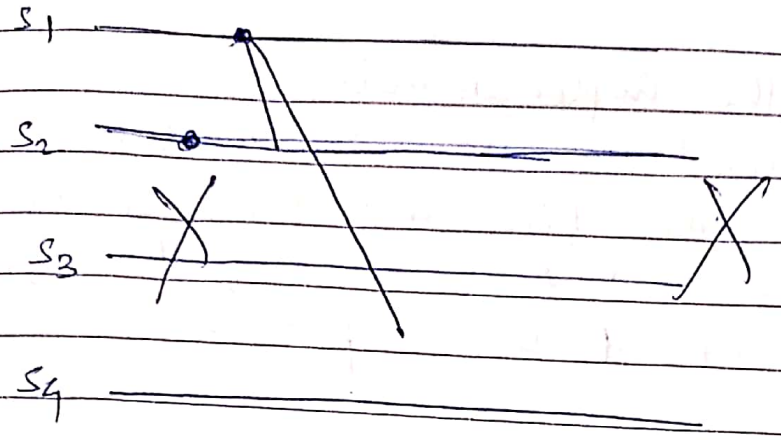
(6) Link the Application

Once all the object code from step 3 and 5 have been compiled, the object implementation class need to be linked to C++ linker. Once linked to the ORB library, two executable operations are created, one for client and one for server

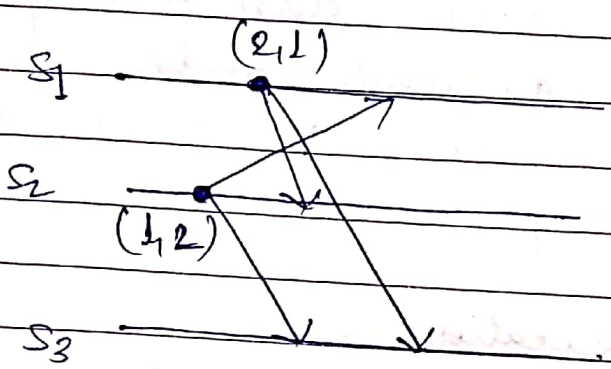
(7) Run the client and server

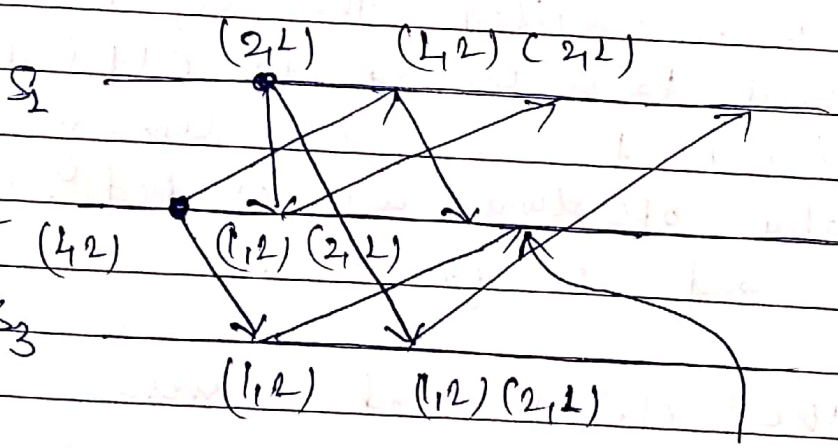
The development process is now complete and the client will now communicate with the server.

Date:
Page:



(Ans-1)

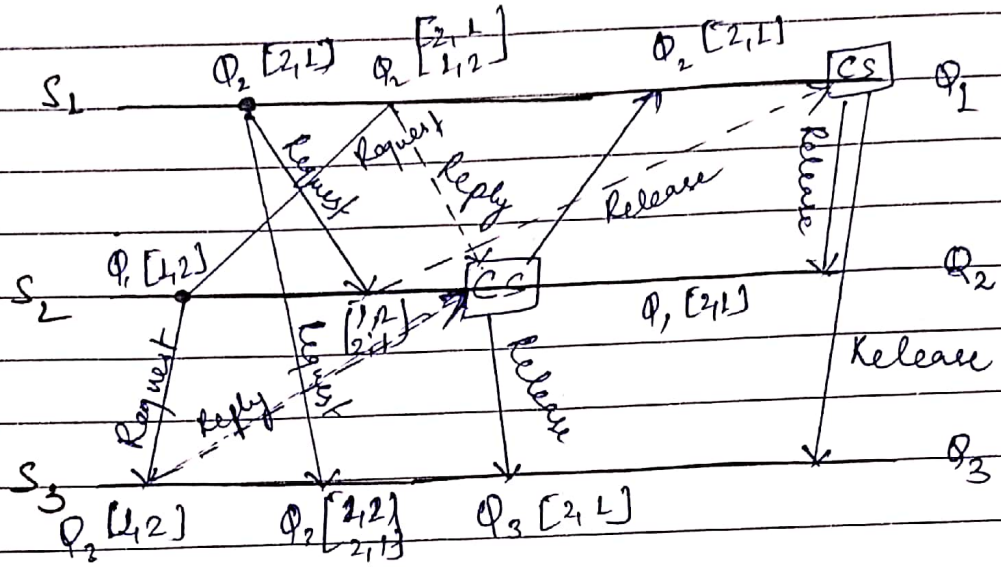




S_2 enters the critical section

Date:
Page:

Using lamport algorithm:-



Request, execute (Reply) and Release states of all these sites are shown above.