Question 6.1 Construct an argument as to why indirect communication may be appropriate in volatile environments. To what extent can this be traced to time uncoupling, space uncoupling or indeed a combination of both?

Answer 6.1

Reason for possibility of indirect communication in volatile environment:

The following is the reason for using the indirect communication in volatile environment:

- The direct communication is not able to deal with the changes in the client-server communication; this is because, in direct communication, it is more complex to replace the alternative server with same functionality.
- Moreover, if the server fails, then it directly affects the client which must obviously fails the communication; but the indirect communication can handle these problems of direct coupling and inherits the properties through space uncoupling and time uncoupling.

Extent of space uncoupling and time uncoupling:

Space uncoupling:

The space uncoupling is that, in which, sender does not know about identify of receiver and receiver does not know about the identity of sender.

Time uncoupling:

In this uncoupling technique, the sender and receiver have separate lifetime; because of this separate lifetime, this does not involve the sender and receiver for the communication at the same time, so, there will not be any communication failure between client and server.

Question 6.2 Section 6.1 states that message passing is both time- and space-coupled – that is, messages are both directed towards a particular entity and require the receiver to be present at the time of the message send. Consider the case, though, where messages are directed towards a name rather than an address and this name is resolved using DNS. Does such a system exhibit the same level of indirection?

Answer 6.2

Explanation:

“Yes”, the above given information also faces the same level of indirection and it is explained as

follows:

- It is a known fact that the DNS name searches for more than one IP address and this is to reduce the workload among the multiple computers.
- With this, it is clear that the name server provides the indirection which means the sender does not exactly bound to the given receiver because the address may be bounded to any one of possible receivers; so, this follows the concept of space uncoupling which is extremely different on communication.
- In space uncoupling, the communication is made explicitly through intermediary such as group multicasting and message queuing.

Therefore, the given case faces same level of indirection.

Question 6.3 Section 6.1 refers to systems that are space-coupled but time- uncoupled – that is, messages are directed towards a given receiver (or receivers), but that receiver can have a lifetime independent from the senders. Can you construct a communication paradigm with these properties? For example, does email fall into this category?

Answer 6.3

Explanation:

With the given case, the communication is possible because the message is directed towards the known identity and it is partially based upon the different lifetime of sender and receiver to make the communication; because, the given case required to have the following assumptions:

Assumption 1:

While making the communication between sender and receiver with the given property rule of space-coupled but time uncoupled, it is interpreted that that receiver is not exist on other side.

Assumption 2:

While making the communication between sender and receiver with the given property rule of space-coupled but time uncoupled, it is interpreted the receiver is available but not available to receive the message on given time.

Question 6.4 As a second example, consider the communication paradigm referred to as queued RPC, as introduced in Rover [Joseph et al. 1997]. Rover is a toolkit to support distributed systems programming in mobile environments where participants in communication may become disconnected for periods of time. The system offers the RPC paradigm and hence calls are directed towards a given server (clearly space-coupled). The calls, though, are routed through an intermediary, a queue at the sending side, and are maintained in the queue until the receiver is available. To what extent is this time uncoupled? Hint: consider the almost philosophical question of whether a recipient that is temporarily unavailable exists at that point in time.

Answer 6.4

Extent of time-uncoupled property:

For the given case, let need to identify up to which extends the time-uncoupled is followed on communication paradigm.

The decision is based upon the point of interpretation of different lifetimes between sender and receiver on communication and the interpretations are as follows:

- If the interpretation is assumed that the receiver is not exists to receive the communication then, this is not the property of time uncoupled.
- If the interpretation is assumed that the receiver is present but not available at the given time which means that the server exists but disconnected due to some problem then, this follows property of time-uncoupled.

Question 6.5 If a communication paradigm is asynchronous, is it also time-uncoupled? Explain your answer with examples as appropriate.

Answer 6.5 Relationship between asynchronous communication and time-uncoupled:

"Yes", if the communication paradigm is asynchronous then it means that it time-uncoupled; because the asynchronous communication and time-uncoupled follows the same rule. The best example, according to given scenario is mail communication and file transfer.

Example #1:

The email communication fits into relationship of both asynchronous communication and time uncoupled.

- If the sender sends a mail which is directed to corresponding receiver who may or may present at the time to receive the message; but, the receiver receives the message when they are connected to mailbox.
- The communication is possible in this case, because the delivery of message is made to corresponding person and it is achieved later.
- Thus, the sender and receiver do not depend upon the time

Questions 6.6 In the context of a group communication service, provide example message exchanges that illustrate the difference between causal and total ordering.

Answer 6.6 The group communication must allow the people to process on different hosts and it must provide support for communication and synchronization between them.

Difference between causal ordering and total ordering:

Causal ordering	**Total ordering**
The casually ordered multicast is a process, which needs to prove the following statement: multicast(g, m)-> multicast(g, m')	In this operation, each process maintains same ordering decision based upon the identifiers.
This contains the symbol "->"and it represents the happened-before relation.	This includes two operations such as "TO multicast" and "TO-deliver".
In group communication, the causal ordering allows the processes once the message "happens before" another message in distributed system.	In group communication, the message must be delivered before another message at one process; so that it follows the same order in all the process.
This casual ordering relationship maintains the result of related messages at all processes.	The total ordering relationship preserves the same order at all processes

Question 6.7 Consider the FireAlarm example as written using JGroups (Section 6.2.3). Suppose this was generalized to support a variety of alarm types, such as fire, flood, intrusion and so on. What are the requirements of this application in terms of reliability and ordering?

Answer:

Requirements in terms ordering:

• For ordering, there are not more important requirements discussed in the "FireAlarm" example, because there is no strong requirement to protect the ordering across alarm types.

• Moreover, this application is simple and there is no sequence of messages related to alarm. Thus, there is no important requirement for ordering.

Question 6.8 Suggest a design for a notification mailbox service that is intended to store notifications on behalf of multiple subscribers, allowing subscribers to specify when they require notifications to be delivered. Explain how subscribers that are not always active can make use of the service you describe. How will the service deal with subscribers that crash while they have delivery turned on?

Use of mailbox service:

If the subscribers are not always active, then the following is the way to make use of the service:

- The service is used by registering the clients with it and the client receives the registration object.
- This object is saved in a file to register the "RemoteEventListener" provided by the mailbox service.
- This creates the event generators for the event that requires the notification; thus, by this way, the notifications are notified later even though the subscribers are not always active in the mailbox service.

Use of mailbox on crashing:

If the mailbox is crashed by client, then the following is the way to provide the service:

- The mailbox can be restored by the registration object when it is restarted.
- So, whenever the mailbox needs to receive the events then the delivery is turned on and if it is not required then it makes the delivery off.

Question 6.9 In publish-subscribe systems, explain how channel-based approaches can trivially be implemented using a group communication service? Why is this a less optimal strategy for implementing a content-based approach?

Answer 6.9

Channel based approach on publish-subscribe system:

This is one of the comfortable approaches for publish-subscribe system.

- This is the scheme which is successfully used in Common Object Request Broker Architecture (CORBA) event service.

- In this approach, the events are published by the publishers to named channels and then subscribers select any one from the named channel to receive all the events posted by publisher.

Reason for less optimal in content based approach:

This content based approach is considered as less optimal strategy because of the following reasons:

- This is more expressive than the channel and topic based approach in publish-subscribe system.
- The query language involved in this content based approach varies from system to system.
- Mapping operation cannot be possible; so, the communication is more associative and content oriented.

Question 6.10 Using the filtering-based routing algorithm in Figure 6.11 as a starting point, develop an alternative algorithm that illustrates how the use of advertisements can result in significant optimization in terms of message traffic generated.

Answer 6.10:

Working method of filtering based routing algorithm:

This is the algorithm in which the broker receives the request and it passes the notification to all connected nodes.

- The match function matches the event from subscription list and then it forwards the event to all the nodes in subscription.
- While using the match function, the event is matched against the routing table and then it forwards path to lead to a subscription.

Advertisement-based approach:

The above filtering based approach creates lot of traffics due to subscriptions and this can be reduced by the advertisement based approach.

• Similar to filtering based approach, the advertising approach follows the same principle to publish the events.

• The only difference is that each node is now supported with the additional advertisement-based routing table.

Question 6.11 Construct a step-by-step guide explaining the operation of the alternative rendezvousbased routing algorithm shown in Figure 6.12.

Answer 6.11

Step-by-step procedure of rendezvous routing algorithm:

This is the approach which contains rendezvous nodes and these nodes referred as broker nodes. This algorithm is achieved only if the following steps are achieved:

- First, the subscription “s” is received on the SN(s) and then it returns one or more rendezvous nodes. These nodes takes the responsibility for the given subscription and additionally, the nodes maintains the list of subscriptions like filtering based approach and based on the list it forwards all the matching events to the subscribing nodes.
- The second step is whenever the event “e” is published, attleast one or more rendezvous nodes is returned by the function EN(e) and it matches the event “e” against subscriptions.

Question 6.12 Building on your answer to Exercise 6.11, discuss two possible implementations of EN(e) and SN(s). Why must the intersection of EN(e) and SN(s) be non-null for a given e that matches s (the intersection rule)? Does this apply in your possible implementations?

Answer 6.12

Possible implementation of " EN(e) " and " SN(s) ":

Rendezvous routing algorithm is one of the approach which control the subscriptions and this contains rendezvous nodes. This algorithm is achieved only if the following steps are achieved:

- First, the subscription "s" is received on the "SN(s)" and then it returns one or more rendezvous nodes. These nodes takes the responsibility for the given subscription and additionally, the nodes maintains the list of subscriptions like filtering based approach and based on the list it forwards all the matching events to set of subscribing nodes.
- The second step is whenever the event "e" is published; it returns the rendezvous nodes and these nodes matches the event "e" against subscriptions.

Reason for the intersection rule:

In rendezvous based routing algorithm, the subscription events are sent to one or more nodes and the publish event routes all the subscription. For any specific event, the nodes calculated and offered by "EN(e)" must overlap with the calculation of "SN(s)" to find the matching node to the subscription and then it is forwarded in the network.

Implementations:

Consider the following two example implementations of rendezvous based routing algorithm:

Example #1:

Assume the first example as implementation of distributed hash table (DHT) which contains publish subscribe system, in which, the "EN(e)" and "SN(s)" are the identical operation which implements the hash function used for topic name.

Application of rendezvous based routing in implementations:

"Yes", both the above implementation applies the rule of intersection and argument rule.

Question 6.13 Explain how the loose coupling inherent in message queues can aid with Enterprise Application Integration. As in Exercise 6.1, consider to what extent this can be traced to time uncoupling, space uncoupling or a combination of both.

Answer 6.13

Way that EAI achieve the loose coupling in message queue:

The EAI is the framework integrated with the required services by the organizations. According to the concept of EAI, the message queues also provide the indirection which means the communication between the services is indirect and they should not communicate with each other but can communicate through message queue.

Extent of message queue to time uncoupling and space uncoupling:

Refer the "exercise 6.1" which describes about the concept of space uncoupling and time uncoupling. According to the space uncoupling, the key is that it does not need to identify the identity of the recipient. Here, the message queue does not know about each other, so, the space uncoupling is suitable to these criteria; but when compared to the concept of time uncoupling, the services available in the message queue is available at any time, so, it is not required to bother about the time to complete the process.

Question 6.14 Consider the version of the FireAlarm program written in JMS (Section 6.4.3). How would you extend the consumer to receive alarms only from a given location?

Answer 6.14

Extending the message consumer to receive the alarm objects:

Actually, the code referred in the "section 6.4.3" is more complicated because it needs to create the connection, publisher, and message.

- Moreover, the parameters to create the topic session and mode of acknowledgement are not clearly specified and also there is some complexity to find out the connection factory.
- But these problems can be resolved through Java Naming and Directory Interface (JNDI).
- Similar to code in figure 6.17 in the textbook, the figure 6.18 creates the connection and session through lines 2 to 5. Next at the line number 10, the object of "TopicSubscriber' is created and "start()" method in line number 11 starts the subscription which makes the messages to be received.

Question 6.15 Explain in which respects DSM is suitable or unsuitable for client-server systems.

Answer 6.15

DSM's suitability for client-server systems:

DSM is often considered incompatible for client-server systems since it is not favorable for working heterogeneously.

Further, with respect to security, client-server systems need a shared region per client; however, for DSM, this would be expensive. In some application domains, DSM may be considered suitable for client-server systems.

- For instance, where a group of clients can typically share server replies.

Question 6.16 Discuss whether message passing or DSM is preferable for fault-tolerant applications.

Answer 6.16

Preference:

For fault-tolerant applications, message passing is preferable over DSM.

Reasons:

Consider couple of processes running at independent computer machines. In a system with message passing,

- When a process has a bug, to some extent the other process may guard itself by authorizing the messages it receives.
- When a process fails half-way through an operation, then some transactional techniques could
- be employed to make sure that data are placed in a very consistent state. In a system with DSM,
- If one process has a bug, then other process may be adversely affected because one process could update a shared-variable without the other's knowledge.

Question 6.17 Assuming a DSM system is implemented in middleware without any hardware support and in a platform-neutral manner, how would you deal with the problem of differing data representations on heterogeneous computers? Does your solution extend to pointers?

Answer 6.17

Dealing of different data representation in DSM system:

• This form of representation means that DSM maintains the layout and types of data which is converted form of local representation.

The page fault in this implementation does not depend upon the machine architecture; if the page sizes are different, then it becomes a big issue of marshalling and unmarshalling because the data items exceeds out of page boundaries.

For this problem, let use the "virtual page" concept because it allows the maximum of page size supported by all the architectures. The data items can also be protected inside the boundary and same of items are maintained in all architectures.

Extend to pointers:

"Yes", the marshaling and unmarshalling method of middleware architecture also marshals the pointers.

• Pointers are marshaled until the kernel forms the layout of the data.

• These pointers express the offset of the data item.

Question 6.18 How would you implement the equivalent of a remote procedure call using a tuple space? What are the advantages and disadvantages of implementing a remote procedure call–style interaction in this way?

Answer 6.18

Implementing the remote procedure call using tuple space:

The way to implement the remote procedure call using the tuple space is as follows:

• The client must write the tuple into tuple space which directly points the server and operation name within this server.

• While writing the tuple, it must include the information about the respective fields such as request, nonce (which means that the request is unique), server name, name of the operation, and parameter list.

o The client now starts to perform the take operation which means it blocks the client until the remote procedure call gets completed.

o Then the server will continue the loop to take the operation by matching the tuple with corresponding server name and it takes other values to perform the operation.

Advantages of implementing the remote procedure using tuple space:

• Even though server is not available at the time of request, the server will manage it by matching the corresponding tuple when it is reconnected with tuple space

• If there are multiple servers matching with the same name, then this system automatically balances the load and also manages the fault tolerance by enabling systems to continue the service.

Disadvantages of implementing the remote procedure call using tuple space:

The following are the disadvantages of implementing the remote procedure call using tuple space:

• There is an additional indirection when the server is not available at the time of request in the application.

• This leads to increase the latency of interaction

Question 6.19 How would you implement a semaphore using a tuple space?

Answer 6.19

Implementation of semaphore using tuple space:

The semaphore using tuple space is implemented by kernel Linda through several methods on both single and distributed memory computers.

- Since the tuple space implementation is mainly concentrated on distributed computing, the following are the approach:
    - The composite objects are maintained using the distributed pointer called as locator.
    - This locator helps to locate the processor, in which, the object is located through process ID.
    - If any of the operation is needed to be performed, then the message is sent to the processor with the object.

Question 6.20 Implement a replicated tuple space using the algorithm of Xu and Liskov [1989]. Explain how this algorithm uses the semantics of tuple space operations to optimize the replication strategy

Answer 6.20

Implementation of replicated tuple space using Xu and Liskov algorithms:

In this approach, “view” is the context which contains agreed collection of replicas and tuples are split into distinct tuple sets based on their logical names.

- This system has set of workers to perform the operation on tuple space and set of tuple space replicas.
- The physical node in this method may contain any number of workers, replicas or both; each worker may or may not contain local replica.
- In a communication network, nodes connected might loss, or delay and it can be delivered out of order.
- In Xu and Liskov algorithms, the following are the operation implemented using tuple space:

    - Write operation:

This operation is implemented over the multicast message on unreliable communication channel to all members. The write request is repeated until it receives the acknowledgement.

- Read operation:

This operation is implemented over the multicast message and it sends this message to all replicas; this operation returns the result of first tuple.

- Take operation:

This operation is more complex because this needs to agree on the tuple for which it is selected

and then it removes agreed tuple from copies