1. SHARING - Give five types of hardware resource and five types of data or software resource that can usefully be shared. Give examples of their sharing as it occurs in practice in distributed systems.
2. SERVICE DISCOVERY - A user arrives at a railway station that she has never visited before, carrying a PDA that is capable of wireless networking. Suggest how the user could be provided with information about the local services and amenities at that station, without entering the station's name or attributes. What technical challenges must be overcome?
3. CLOUD COMPUTING DEFINITION - Compare and contrast cloud computing with more traditional client-server computing? What is novel about cloud computing as a concept?
4. WWW - Use the World Wide Web as an example to illustrate the concept of resource sharing, client and server. What are the advantages and disadvantages of HTML, URLs and HTTP as core technologies for information browsing? Are any of these technologies suitable as a basis for client-server computing in general?
5. LOCATION TRANSPARENCY - Give an example of an HTTP URL. List the main components of an HTTP URL, stating how their boundaries are denoted and illustrating each one from your example. To what extent is an HTTP URL location-transparent?
6. FAILURE - List the three main software components that may fail when a client process invokes a method in a server object, giving an example of a failure in each case. Suggest how the components can be made to tolerate one another's failures.
7. REPLICATION - A service is implemented by several servers. Explain why resources might be transferred between them. Would it be satisfactory for clients to multicast all requests to the group of servers as a way of achieving mobility transparency for clients?
8. CLIENT-SERVER ARCHITECTURE - Describe and illustrate the client-server architecture of one or more major Internet applications (for example the Web, email or netnews)
9. SYNCHRONISATION - A search engine is a web server that responds to client requests to search in its stored indexes and (concurrently) runs several web crawler tasks to build and update the indexes. What are the requirements for synchronisation between these concurrent activities?
10. REPLICATION and QOS - The host computers used in peer-to-peer systems are often simply desktop computers in users' offices or homes. What are the implications of this for the availability and security of any shared data objects that they hold and to what extent can any weaknesses be overcome through the use of replication?
11. SECURITY - List the types of local resource that are vulnerable to an attack by an untrusted program that is downloaded from a remote site and run in a local computer
12. MOBILITY - Give examples of applications where the use of mobile code is beneficial.
13. CLIENT-SERVER PERFORMANCE- Consider a simple server that carries out client requests without accessing other servers. Explain why it is generally not possible to set a limit on the time taken by such a server to respond to a client request. What would need to be done to make the server able to execute requests within a bounded time? Is this a practical option?
14. CLOCK SYNCHRONISATION - The Network Time Protocol service can be used to synchronize computer clocks. Explain why, even with this service, no guaranteed bound is given for the difference between two clocks.
15. COMMUNICATION RELIABILITY - Consider two communication services for use in asynchronous distributed systems. In service A, messages may be lost, duplicated or delayed and checksums apply only to headers. In service B, messages may be lost, delayed or delivered too fast for the recipient to handle them, but those that are delivered arrive with the correct contents. Describe the classes of failure exhibited by each service. Classify their failures according to their effect on the properties of validity and integrity. Can service B be described as a reliable communication service?
16. FAILURE TYPES - Suppose that a basic disk read can sometimes read values that are different from those written. State the type of failure exhibited by a basic disk read. Suggest how this failure may be masked in order to produce a different benign form of failure. Now suggest how to mask the

benign failure

17. COMMUNICATION INTEGRITY - Define the integrity property of reliable communication and list all the possible threats to integrity from users and from system components. What measures can be taken to ensure the integrity property in the face of each of these sources of threats.

18. COMMUNICATION SECURITY - Describe possible occurrences of each of the main types of security threat (threats to processes, threats to communication channels, denial of service) that might occur in the Internet

19. SCALING - The Internet is far too large for any router to hold routing information for all destinations. How does the Internet routing scheme deal with this issue?

———

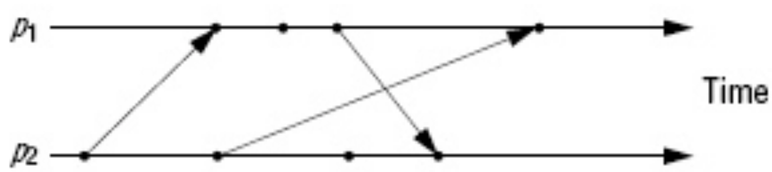
1. MUTUAL EXCLUSION - In the central server algorithm for mutual exclusion, describe a situation in which two requests are not processed in happened-before order.

2. MUTUAL EXCLUSION and FAULTS - Adapt the central server algorithm for mutual exclusion to handle a crash failure of any client in any stat. Assume that the server is correct and given a reliable failure detector. Is the resultant system is fault tolerant? What if a client with the token is wrongly suspected to have failed?

3. BULLY ALGORITHM - Suggest how to adapt the bully algorithm to deal with temporary network partitions (slow communication) and slow process

4. SNAPSHOT ALGORITHM - two processes p_1 *and* p_2 communicate in a ring using two channels constantly rotating a message *m*. At any one time, there is only one copy of *m* in the system. Each process's state consists of the number of times it has received *m*, and p_1 sends *m* first. Then p_1 has the message and its state is 101, and immediately after sending *m*, p_1 initiates the snapshot algorithm. Explain the operation of the algorithm in this case, giving the possible global state(s) reported by it.

5. CONSISTENT STATE - Draw and label the lattice of consistent states (*p1* state, *p2* state), beginning with the initial state (0,0)



6. CLOCK SYNCHRONIZATION - Why is computer clock synchronization necessary? What are the design requirements for clock synchronization s in a distributed system?

7. CONSISTENT STATE - A client tries to synchronize with a time server. It records the round-trip times and timestamps returned by the server the values *Round-trip (ms) Time (hr:min:sec) 22 10:54:23.674 25 10:54:25.450 20 10:54:28.342* Which of these times should it use to set its clock? To what time should it set it? How can you estimate the accuracy of the setting with respect to the server's clock, knowing that the time between sending and receiving a message in the system is at least 8 ms?

8. NTP - An NTP server B receives server A's message at 16:34:23.480, bearing a timestamp of 16:34:13.430, and replies to it. A receives the message at 16:34:15.725, bearing B's timestamp, 16:34:25.7. Estimate the offset between B and A, and the accuracy of the estimate.

———-

1. TRANSACTION INTERLEAVING - A server manages objects a_1, a_2, ... a_n. And provides operations *read* (*i*) returns the value of *ai write*(*i, value*) assigns *value* to a_i Give three serially equivalent interleavings of the following transactions *T* and *U T*: *x* = *read*(*j*); *y* = *read* (*i*); *write*(*j*, 44); *write*(*i*, 33); *U*: *x* = *read*(*k*); *write*(*i*, 55); *y* = *read* (*j*); *write*(*k*, 66)

2. SERIAL EQUIVALENCE - give serially equivalent interleavings of *T* and *U* of the previous exercise, in the three cases: 1) strict; 2) not strict, but could not produce cascading aborts; 3) that

could produce cascading aborts.

3. NESTING – Consider transactions T and U: *T*: a.withdra(4); b.deposit(4); *U*: c.withdraw(3); b.deposit(3); Assume that they are nested as follows: *T*1: a.withdraw(4); *T*2: b.deposit(4); *U*1: c.withdraw(3); *U*2: b.deposit(3); Compare the number of serially equivalent interleavings of *T*1, *T*2, *U*1 and *U*2 with the number of serially equivalent interleavings of *T* and *U*. Why nested transactions generally allow a larger number of serially equivalent interleavings than non-nested ones? Explain.

4. LOCKING - Why serial equivalence requires that once a transaction has released a lock on an object, it is not allowed to obtain any more locks? Explain. A server manages objects a_i, $1 \le i \le n$. with two operations *read*(*i*) and *write*(*i*, *Value*) Give an example of interleaving of transactions *T* and *U* that is not serially equivalent because locks are released early. *T*: *x* = *read*(*i*); *write*(*j*, 44); *U*: *write*(*i*, 55); *write*(*j*, 66);

5. TWO-PHASE LOCKING - Consider a variant of the two-phase locking where read-only transactions can release read locks early. Would a read-only transaction have consistent retrievals? Would the objects become inconsistent? Illustrate your answer with the following transactions *T* and *U* at the server in exercise 4, with initial values a_i =10 and a_j =20. *T*: *x* = *read* (*i*); *y*= *read*(*j*); *U*: *write*(*i*, 55); *write*(*j*, 66);

6. TWO-PHASE LOCKING Consider transactions *T* and *U,* with initial values a_i =10 and a_j =20. *T*: (t1) *x* = *read*(*i*); (t2) *write*(*j*, 44); *U*: (u1) *write*(*i*, 55); (u2) *write*(*j*, 66); Which of the following interleavings are serially equivalent, and which could occur with two-phase locking? a) t1, u1, t2, u2 c) u1, u2, t1, t2 b) t1, t2, u1, u2 d) u1, t1, u2, t2

7. OPTIMISTIC METHOD Consider the optimistic concurrency control applied to transactions of exercise 6, by assuming that transactions *T* and *U* are active at the same time. Describe the outcome in each of the following cases: *i) T*'s request to commit comes first and backward validation is used *ii) U*'s request to commit comes first and backward validation is used *iii) T*'s request to commit comes first and forward validation is used iv) *U*'s request to commit comes first and forward validation is used Describe the sequence of operations of *T* and *U*. Note that writes are not carried out until after validation.

8. TIME STAMP Consider the use of timestamp ordering with each of the example interleavings of transactions T and U in Exercise 6, with initial values a_i =10 and a_j =20, and initial *read* and *write* timestamps are t_0. Assume that each transaction opens and obtains a timestamp just before its first operation; for example, in (a) *T* and *U* get timestamps t_1 and t_2 respectively, where $t_0 < t_1 < t_2$. Describe in order of increasing time the effects of each operation of *T* and *U*. For each operation, state the following: i) whether the operation may proceed according to the write or read rule; ii) timestamps assigned to transactions or objects; iii) creation of tentative objects and their values. What are the final values of the objects and their timestamps?

——

1. AVAILABILITY - A service is provided by four computers. If the mean time between failure of each computer is of five days and a failure typically takes four hours to fix. What is the availability of the replicated service?

2. VIEW SYNVHRONOUS – Two processes *q* and *r* are connected with a router that fails immediately after *p* initiates the multicasting of message *m*. If the group communication system is view-synchronous what happens to *p* next?

3. CAUSAL ORDERING – An operation *X* upon an object *o* causes *o* to invoke an operation upon another object *o'.* Now consider the proposal to replicate *o* but not *o*'. Explain the possible problem concerning invocations upon *o*', and suggest a solution.

4. CONSISTENCY TYPES – Explain the difference between linearizability and sequential consistency, and why the latter is more practical to implement, in general.

5. GOSSIP - Explain why making some replica managers read-only may improve the performance of a gossip system.

6. DISTRIBUTED FILE SYSTEM –Why is there no *open* or *close* operation in the interface to the flat

file service or the directory service? What are the differences between the considered directory service *Lookup* operation and the UNIX *open?*

7. FILE IDENTIFIER – An operation *X* upon an object *o* causes *o* to invoke an operation upon another object *o'.* Now consider the proposal to replicate *o* but Why should UFIDs be unique across all possible file systems? How is uniqueness for UFIDs ensured?

8. MOUNTING – How does the NFS automounter help to improve the performance and scalability of file system NFS?

---

1. CLOUD RESOURCES – Give examples of cloud resources, physical resources and logical resources. Describe how they are provided in cloud computing.

2. SCALING – Give two examples of application of vertical scaling and horizontal scaling in cloud computing. Discuss the main differences between the two techniques.

3. SERVICE DELIVERY MODEL– Describe how the various types of service delivery model provide services in cloud computing, what are the main activities of cloud user and cloud provider.

4. DEPLOYMENT MODELS – Describe the main deployment models in cloud computing, how and when the access to cloud can be limited. Discuss the security management implementation of the various models.

5. ENABLING TECHNOLOGIES – Discuss why and how the advances in web technology and web-based application architecture are related to the development of cloud computing.

6. ELASTICITY– Discuss elasticity in cloud computing and how can be measured to provide rapid reaction in resource demand management.

7. SERVICE LEVEL AGREEMENT– Discuss the importance of SLA in cloud computing, the advantages and the possible management difficulties.

8. BOUNDARIES AND SECURITY – Discuss the relation between security and boundary of cloud computing systems. Consider the case of a cloud consumer that accesses a cloud service of the same organization X, that in turn access a service of public organization Y. What is the relation of boundaries and security?