DISTRIBUTED SYSTEMS

B.Tech (C.S.E), 7th Semester (A.Y. 2022-23) NIT Raipur

Questions Bank

UNIT-I

1. In what respect are distributed computing systems better than parallel processing systems? Give examples of three applications for which distributed computing systems will be more suitable than parallel processing systems.
2. What were the major technological, economical, and social factors that motivated the development of distributed computing systems? What are some of the main advantages and disadvantages of distributed computing systems over centralized ones?
3. Discuss the relative advantages and disadvantages of the various commonly used models for configuring distributed computing systems. Which model do you think is going to become the most popular model in future? Give reasons for your answer.
4. Draw the Omega and Butterfly networks for n = 16 inputs and outputs.
5. Formulate the interconnection function for the Omega network having n inputs and outputs, only in terms of the M = n/2 switch numbers in each stage. (Hint: Follow an approach similar to the Butterfly network formulation.)
6. Explain why a Receive call cannot be asynchronous.
7. Describe Message-passing systems versus shared memory systems.
8. Describe Emulating the message-passing on a shared memory system versus Emulating the shared memory on a message-passing system.
9. Explain the difference between the terms service and server. In the design of a distributed operating system, discuss the relative advantages and disadvantages of using a single server and multiple servers for implementing a service.
10. Why are distributed operating systems more difficult to design than operating systems for centralized time-sharing systems?
11. What are the main differences between a network operating system and a distributed operating system?
12. What are the major issues in designing a distributed operating system?
13. A distributed operating system makes a collection of networked machines to act like a virtual Uni-processor. What are the main advantages of this virtual-machine architecture for a user? What issues are important for a distributed operating system designer in achieving this goal?
14. Discuss some of the important concepts that a distributed operating system designer might use to improve the reliability of his or her system. What is the main problem in making a system highly reliable?
15. Discuss the main guiding principles that a distributed operating system designer must keep in mind for the good performance of his or her system.

UNIT-II

Remote Procedure Calls (RPC)

1. What was the primary motivation behind the development of the RPC facility? How does an RPC facility make the job of distributed applications programmers simpler?
2. In the conventional procedure call model, the caller and the callee procedures often use global variables to communicate with each other. Explain why such global variables are not used in the RPC model.
3. What are the main issues in designing a transparent RPC mechanism? Is it possible to achieve complete transparency of an RPC mechanism? If no, explain why. If yes, explain how.
4. A server is designed to perform simple integer arithmetic operations (addition, subtraction, multiplication, and division). Clients interact with this server by using an RPC mechanism. Describe the contents of the call and reply messages of this RPC application, explaining the purpose of each component. In case of an error, such as division by zero or arithmetic overflow, the server must suitably inform the client about the type of error.
5. Differentiate between stateful and stateless servers. Why do some distributed applications use stateless servers in spite of the fact that stateful servers provide an easier programming paradigm and are typically more efficient than stateless servers?
6. Discuss the similarities and differences between the following parameter-passing semantics that may be used in an object-based system: (a) Call-by-object-reference, (b) Call-by-move, (c) Call-by-visit
7. Explain why RPC semantics is normally different from the conventional procedure call semantics. Clarify the differences among may-be, last-one, last-of-many, at-least-once, and exactly-once call semantics. Explain how each of these may be implemented.
8. Explain why most RPC systems do not use acknowledgment messages. Differentiate among R, RR, and RRA protocols for RPCs. Give an example of an application in which each type of protocol may be the most suitable one to use.
9. What are the main advantages of an RPC system that allows the binding between a client and a server to change dynamically? What are the main issues involved in providing this flexibility? Describe a mechanism to handle each of the issues mentioned by you.
10. Discuss the relative advantages and disadvantages of binding a client and a server at compile time, at link time, and at call time.
11. Given the interface name of a server, discuss the relative advantages and disadvantages of using the broadcast method and the method of using a name server for locating the server.
12. What is callback-RPC facility? Give an example of an application where this facility may be useful. What are the main issues involved in supporting this facility in an RPC system? Describe a mechanism to handle each of these issues.

Synchronization

13. What was the primary motivation behind the development of the Lightweight RPC (LRPC) system [Bershad et al. I990]? Describe some of the techniques used in the LRPC system that makes it more efficient than conventional RPC systems.

14. Explain why one-time synchronization of the clocks of all the nodes of a distributed system is not sufficient and periodic resynchronization is necessary. How will you determine the interval for periodic resynchronization?

15. A distributed system has three nodes N., N2, and N3, each having its own clock. The clocks of nodes N), N2, and N3 tick 800,810, and 795 times per millisecond. The system uses the external synchronization mechanism, in which all three nodes receive the real time every 30 seconds from an external time source and readjust their clocks. What is the maximum clock skew that will occur in this system?

16. Differentiate between internal synchronization and external synchronization of clocks in a distributed system. Externally synchronized clocks are also internally synchronized, but the converse is not true. Explain why.

17. In distributed systems, there may be unpredictable variation in the message propagation time between two nodes. Explain why. How does this problem make the task of synchronizing clocks in a distributed system difficult? Give two methods that can be used in a clock synchronization algorithm to handle this problem.

18. Explain the concept of logical clocks and their importance in distributed systems. A clock of a computer system must never run backward. Explain how this issue can be handled in an implementation of the logical clocks concept.

19. In the centralized approach to mutual exclusion described in this chapter, the coordinator grants permission for critical section entry to the first process in the queue. In some systems, it may be desirable to grant permission to some higher priority jobs before other lower priority jobs. Modify the algorithm to take care of this and show how your algorithm satisfies the No-starvation property.

20. The first general algorithm for implementing mutual exclusion in a distributed environment was developed by Lamport (1978). Find the details of this algorithm and compare its performance and reliability with that of Ricart and Agrawala's [1981] algorithm.

21. What is a "deadlock"? What are the four necessary conditions for a deadlock to occur? Give suitable examples to prove that if anyone of the four conditions is absent, no deadlock is possible.

22. Prove that for a system having only one unit of each resource type the presence of a cycle in a resource allocation graph is both a necessary and a sufficient condition for the existence of deadlock.

23. What problems may arise when a process is killed/rolled back and then restarted as the result of a deadlock? Suggest suitable methods to handle these problems. What conclusions can be drawn from these problems in connection with the proper selection of victims for recovery from a deadlock state?

24. Why election algorithms are normally needed in a distributed system? A LAN-based distributed system has broadcast facility. Suggest a simple election algorithm for use in this system.

25. Initiation of an election is actually needed only when the current coordinator process fails. However, this is not the case in the bully algorithm, in which an election is also initiated whenever a failed process recovers. Is this really necessary? If yes, explain why. If no, suggest a modification to the bully algorithm in which an ejection is initiated only when the current coordinator fails.

26. In the ring-based election algorithm described in this chapter, a unidirectional ring was used. Suppose the ring is bidirectional. Can the election algorithm be made more efficient? If no, explain why. If yes, suggest such an algorithm and compare the number

of messages needed for electing a coordinator in the two algorithms, assuming that there are n processes in the system.

27. In the ring-based election algorithm described in this chapter, two or more processes may almost simultaneously discover that the coordinator has crashed and then each one may circulate an election message over the ring. Although this does not cause any problem in the election, it results in waste of network bandwidth. Modify the algorithm so that only one election message circulates completely round the ring and others are detected and killed as soon as possible.

28. What will happen in a bully algorithm for electing a coordinator when two or more processes almost simultaneously discover that the coordinator has crashed?