Register Number | | | | | | | | | |

**Sri Sivasubramaniya Nadar College of Engineering, Kalavakkam – 603 110**

(An Autonomous Institution, Affiliated to Anna University, Chennai)

Department of Computer Science and Engineering

**Continuous Assessment Test – III**

**Answer Key**

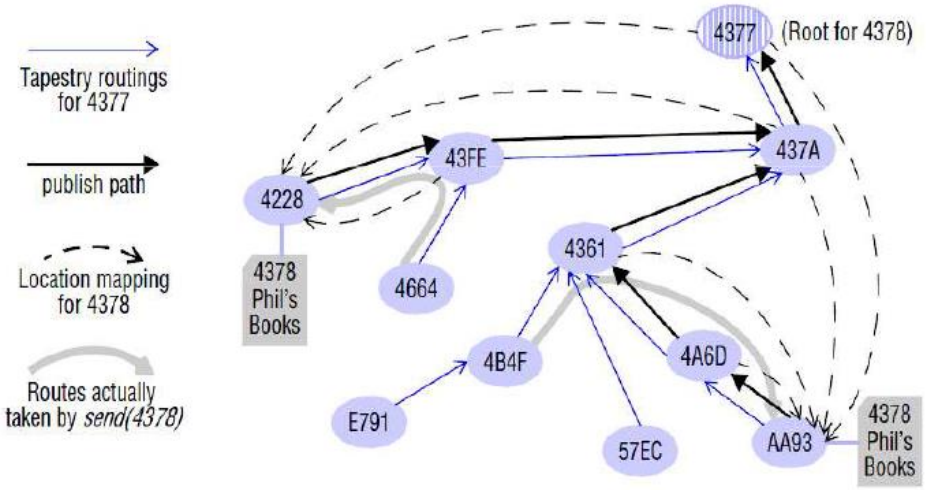
| | | | | | | |
|---|---|---|---|---|---|---|
| **Degree & Branch** | BE & Computer Science and Engineering | | | | **Semester** | VII |
| **Subject Code & Name** | UCS1701- Distributed Systems | | | | **Regulation:** | **2018** |
| **Academic Year** | 2022-2023 ODD | **Batch** | 2019-2023 | **Date** | **12-11-2022** | **FN / ~~AN~~** |
| **Time: 08:15 – 09:45 AM (90 Minutes)** | **Answer All Questions** | | | | **Maximum: 50 Marks** | |

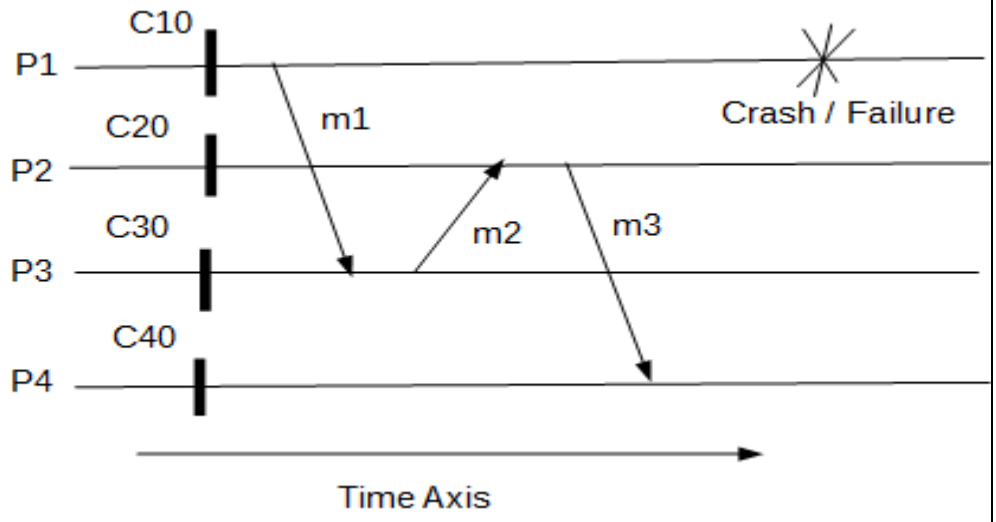
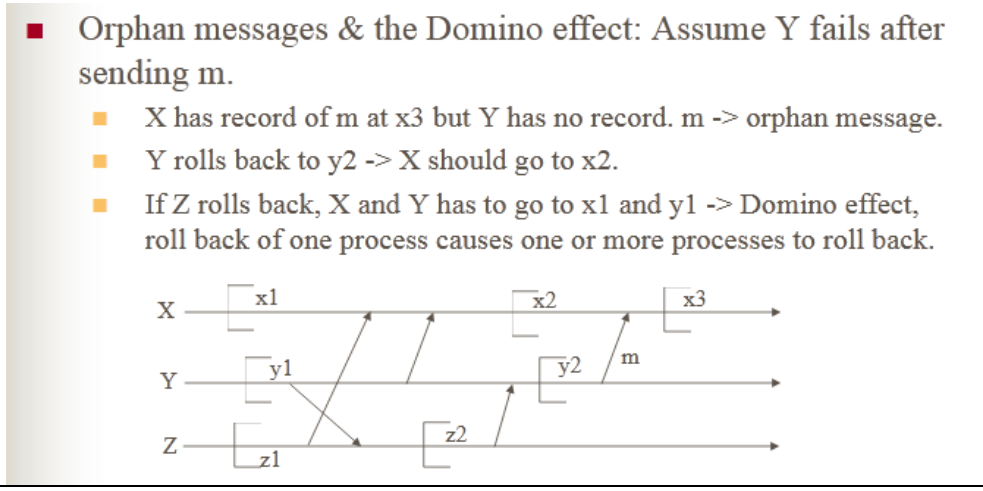
**Part – A (6×2 = 12 Marks)**

| | | |
|---|---|---|
| KL2 | 1. Outline the impact of lost messages on consistency and performance issues.<br>Ans: The lost message in the distributed systems will not affect the consistency of the model, but it would affect the performance of the system. | CO4 |
| KL1 | 2. Define Orphaned messages.<br>Ans: The message received is recorded by receiving process, but the send event is not recorded in the sender process. | CO4 |
| KL1 | 3. Define Overlay Network.<br>Ans: A virtual network formed on top of P2P network is called an overlay network. A network formed on top of the P2P system connecting only nodes which are responsible for the request being processed | CO5 |
| KL2 | 4. Outline the disadvantages of using Global State Recording Protocol for Checkpointing.<br>Ans: GSRP is used to record the global state of the system including the processes and communication channel states. This ensure the consistency of the system but unnecessary checkpoints are taken to ensure global consistency | CO4 |
| KL1 | 5. What is log-based checkpointing?<br>**Ans: Log-based Checkpointing**: Log-based checkpoint assumes that all non-deterministic events can be identified and their corresponding determinants can be logged into the stable storage.<br>Log-based rollback-recovery protocols guarantee that **upon recovery** of all failed processes, the **system does not contain any orphan process.** | CO4 |
| KL2 | 6. Compare stable log with volatile log.<br>Ans: Stable log is non-volatile where the contents of log remain even power is off<br>Volatile log is an unstable log where the contents get erased once power off. | CO4 |

**Part – B (3×6 = 18 Marks)**

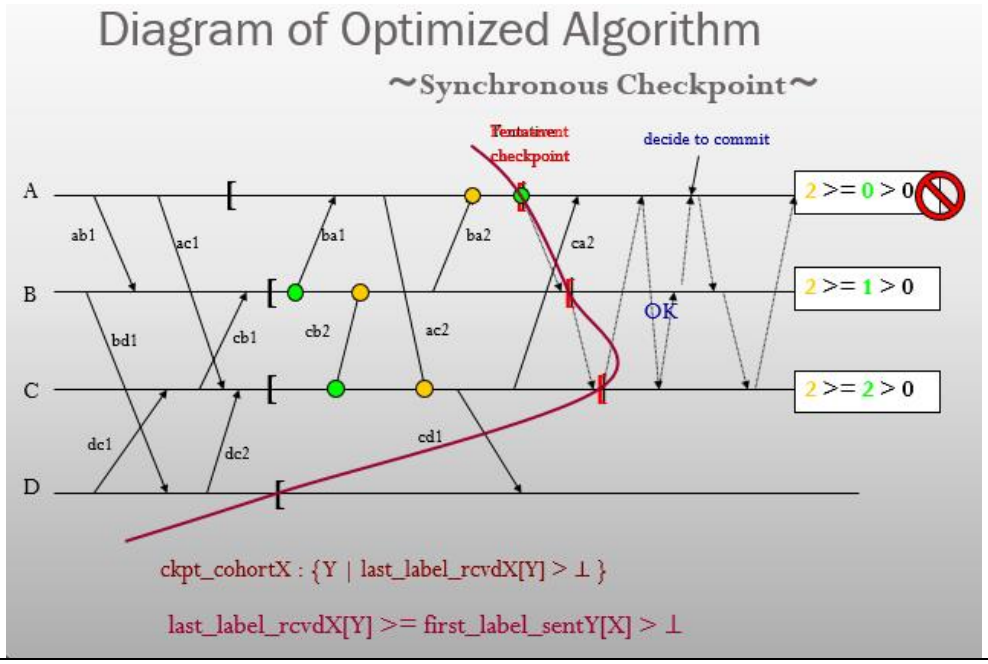
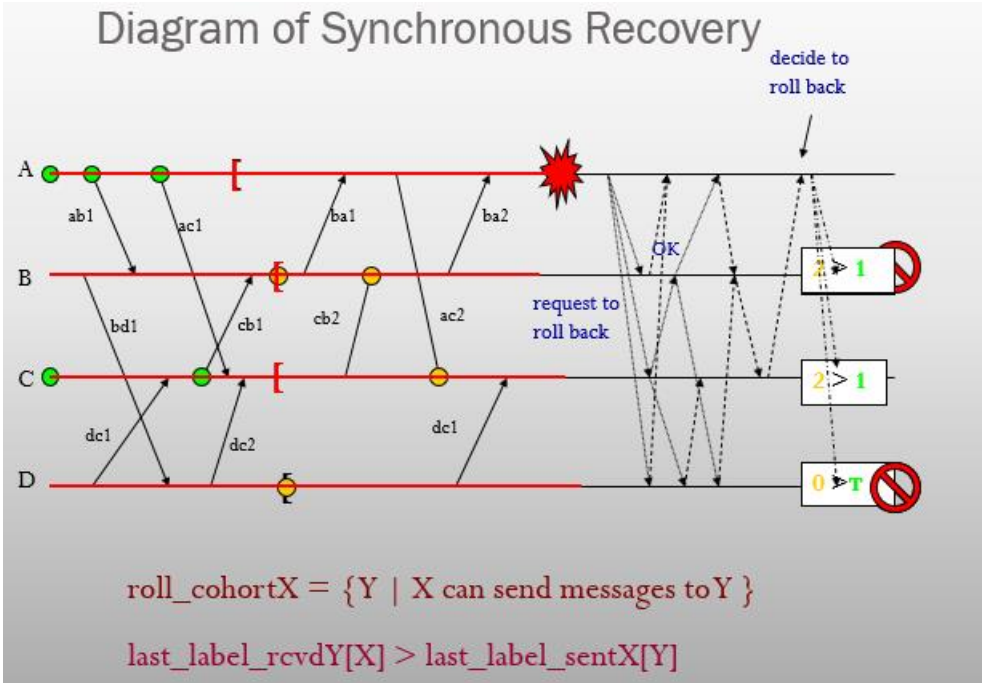
| | | |
|---|---|---|
| KL2 | 7. Explain Peer to Peer systems. Outline the difference between the P2P and client server model.<br>Ans: P2P network: application-level organization of the network to explicitly share resources<br>All nodes are equal; communication directly between peers (no client-server) | CO5 |

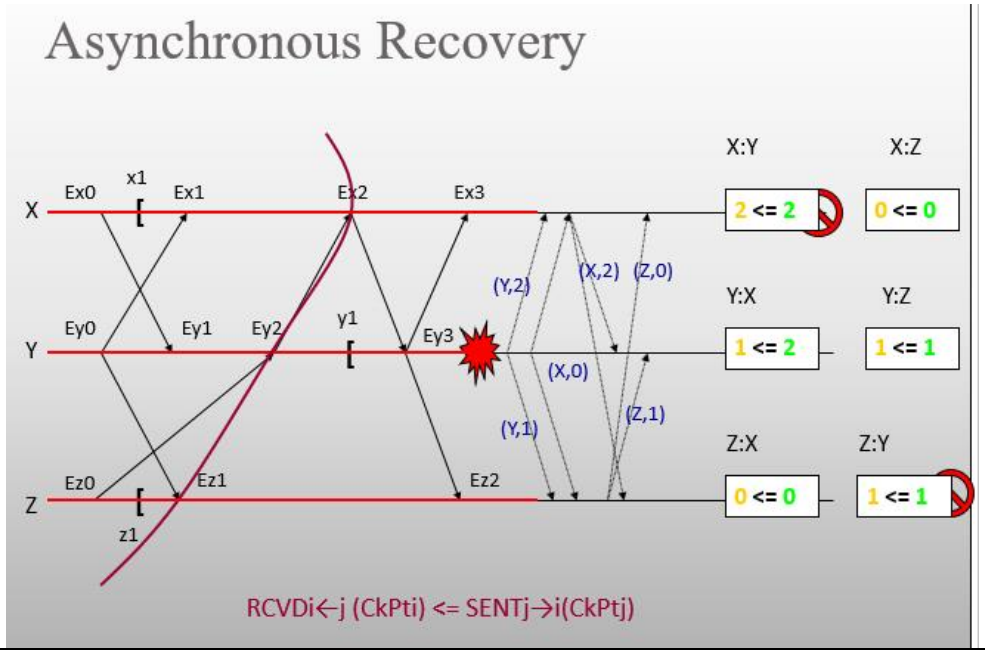
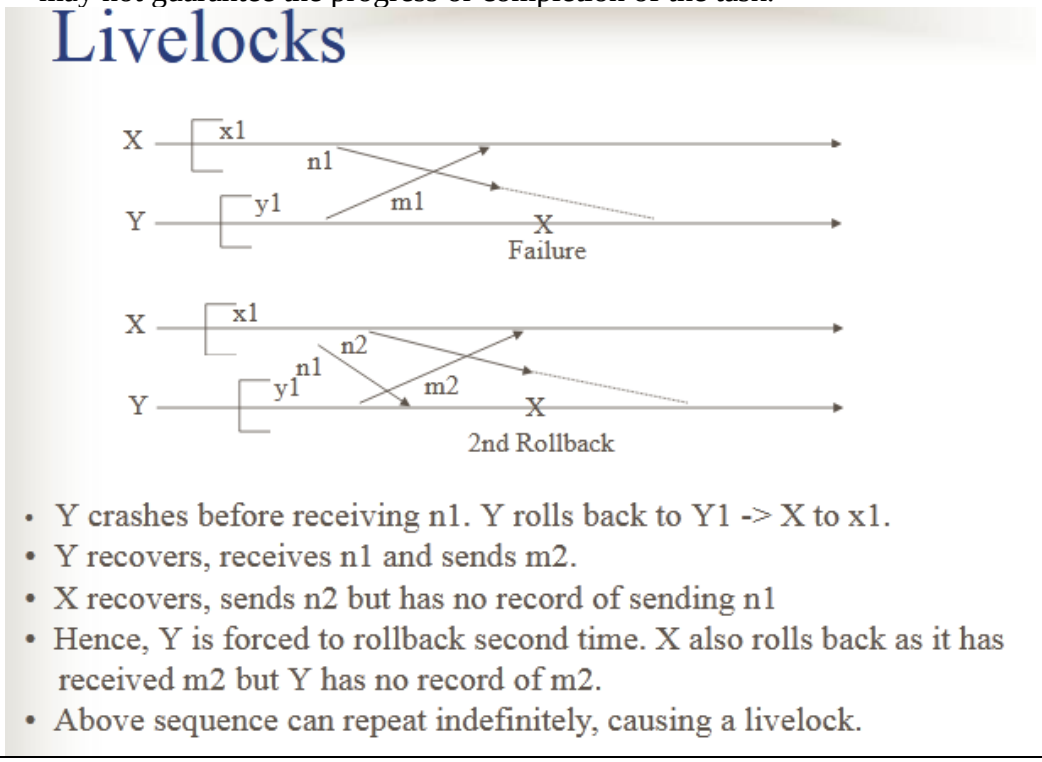
| | | |
|---|---|---|
| | Allow location of arbitrary objects; no DNS servers required<br>Large combined storage, CPU power, other resources, without scalability costs<br>Dynamic insertion and deletion of nodes, as well as of resources, at low cost<br>self-organizing, distributed control, anonymity.<br><br>P2P- No centralized server and client server architecture. Designed to share files between peers. Each peer in the P2P system will process the request from other peers. self-organizing and distributed control<br>Client Server Model: Centralized server and requests are processed by the server. Centralized control. | |
| KL2 | 8. Discuss the routing mechanism followed in Tapestry P2P system.<br>Ans: Tapestry implements a distributed hash table and routes messages to nodes based on GUIDs associated with resources using prefix routing in a manner similar to Pastry.<br>Tapestry applications give additional flexibility:<br>They can place replicas close (in network distance) to frequent users of resources in order to reduce latency and<br>Minimize network load or to ensure tolerance of network and host failures.<br><br>**Distributed Object Location and Routing in Tapestry**<br><br>***publish(GUID)***<br>- *GUID can be computed from the object (or some part of it, e.g. its name).*<br>- This function makes the node performing a *publish operation as host for* the object corresponding to *GUID.*<br>***unpublish(GUID)***<br>- Makes the object corresponding to *GUID inaccessible.*<br>***sendToObj(msg, GUID, [n])***<br>- Following the object-oriented paradigm, an invocation message is sent to an object in order to access it. This might be a request to open a TCP connection for data transfer or to return a message containing all or part of the object's state. The final optional parameter *[n], if present, requests* the delivery of the same message to n replicas of the object.<br><br>**Distributed Object Location and Routing in Tapestry**<br><br>  <br> | CO5 |
| KL3 | 9. Consider 4 processes *A, B, C & D*, develop a scenario which demonstrate Domino's effect. | CO4 |

| | | |
|---|---|---|
| | Ans:<br>  <br><br>When P1 crashes, P1 rollbacks to checkpoint C10 which will make m1 as orphan message at P3. So P3 rollback to C3, m2 becomes orphan at P2, P2 rollback to C20, m3 becomes orphan at P4, P4 rollback to C40. This leads to domino effect.<br>  <br> | |

**Part – C (2×10 = 20 Marks)**

| | | |
|---|---|---|
| KL3 | 10. Consider a suitable topology with 4 processes *A, B, C & D* and assume process *A* initiates the task. Apply **optimized coordinated checkpointing algorithm** for the above given scenario and demonstrate the actions through necessary diagrams. Include messages wherever necessary. | CO4 |

| | | |
|---|---|---|
| | Ans:<br>  | |
| | (OR) | |
| KL3 | 11. Consider a suitable topology with 4 processes *A, B, C & D* and assume process *A* crashes. Apply **synchronous recovery algorithm** for the above given scenario and demonstrate the actions through necessary diagrams. Include messages wherever necessary.<br>Ans:<br>  | CO4 |
| KL3 | 12. Consider a suitable topology with 4 processes *A, B, C & D* and assume process *A* crashes. Apply **asynchronous recovery algorithm** for the above given scenario and demonstrate the actions through necessary diagrams. Include messages wherever necessary. | CO4 |

| | | |
|---|---|---|
| | Ans:<br>  | |
| (OR) | | |
| KL3 | 13. Consider two processes *A & B* which are programmed to send messages to each other parallely. Process *A* sends a message to *B* and crashes. Process *B* message gets delayed and delivered. Identify the issue that affects the consistency and simulate the simulate the sequence of actions for the given scenario.<br>Livelock: **Live Lock:** The processes move from one state to another or messages are exchanged **between two processes** without showing any progress in the task. This may not guarantee the progress or completion of the task.<br>  <br>• Y crashes before receiving n1. Y rolls back to Y1 -> X to x1.<br>• Y recovers, receives n1 and sends m2.<br>• X recovers, sends n2 but has no record of sending n1<br>• Hence, Y is forced to rollback second time. X also rolls back as it has received m2 but Y has no record of m2.<br>• Above sequence can repeat indefinitely, causing a livelock. | CO4 |

**---------- ALL THE BEST----------**