SPC 2401-DISTRIBUTED SYSTEM GROUP

ASSIGNMENT

GROUP MEMBERS

1. NGUGI SYLVIA	**PA106/G/15024/21.**
2. WANJOHI JOHN	**PA106/G/14994/21.**
3. KEPHAR MOSES	**PA106/G/15026/21.**
4. KABINGU MERCY	**PA106/G/15036/21**
5. EVERLYN MUTHONI	**PA106/G/14952/21**
6. ERICK NGECHU	**PA106/G/14972/21**
7. LABAN BETT	**PA106/G/15043/21**
8. JOSEPH MBUGUA	**PA106/G/8524/20**

1. Choose ONE of the following activities and describe the activities associated with it, detailing the process of distribution. [6 Marks]

I)Distributing tasks

This refers to the process of assigning and allocating specific activities or responsibilities among individuals or teams to accomplish a common goal or project. This activity typically takes place within an organization or a group setting, where each member's skills, expertise, and availability are considered for effective task distribution.

The process of distributing tasks usually involves the following steps:

1. Task Identification: First, the project or goal is broken down into smaller tasks or activities and these tasks are identified based on their significance, complexity, and interdependencies.

2. Assessment of Skills and Resources: This involves evaluating the skills, experience, and availability of each team member. This assessment helps determine their suitability and capability to handle specific tasks.
3. Task Assignment: Once the tasks and team members' capabilities are known, the tasks are distributed among the team members. The assignment may depend on various factors, such as individual strengths, interests, expertise, workload balance, and availability.
4. Communication and Instructions: Clear communication is crucial for effective task distribution. The person assigning the tasks must provide detailed instructions and guidelines, including the desired outcome, deadline, and any specific requirements or constraints associated with the task.
5. Monitoring and Support: Throughout the task distribution process, it is important to monitor the progress of assigned.

2. What is a Middleware? What is its key role in Distributed Systems? [2 Marks]

Middleware is a software layer that sits between the operating system and the application software in a distributed computing system.

It's key role in distributed systems is to enable communication and coordination between different components or services across multiple nodes or machines. Middleware provides a set of services and abstractions that abstract the complexities of distributed computing and allow applications to interact with each other seamlessly.

3. In reference to Client-Server Architecture: Distinguish between the blocking and non-blocking message passing semantics. [4 Marks]

In client-server architecture, the blocking and non-blocking message passing semantics represent different approaches to how a client and server interact and communicate with each other.

1. Blocking Message Passing:

- In blocking message passing semantics, the client sends a message or request to the server and then waits for a response before proceeding further.

- The client thread is blocked until it receives a response from the server.

- This means that the client cannot perform any other tasks or operations until it receives the response.

- Blocking message passing ensures that the client waits until it gets a valid response from the server, ensuring synchronization and reliability.

2. Non-blocking Message Passing:

- In non-blocking message passing semantics, the client sends a message or request to the server but does not wait for a response immediately.

- The client thread is not blocked and can continue to perform other tasks or operations while waiting for a response.

- The client may periodically check for a response from the server or use callbacks, events, or interrupts to receive the response asynchronously.

- Non-blocking message passing allows the client to be more responsive and efficient, as it can continue executing other tasks without being stalled.

Overall, the key difference between blocking and non-blocking message passing semantics lies in how the client handles waiting for a response from the server.

4. State and explain the three semantics for reliable IPC protocol. [6 Marks]

1. At-most-once

The at-most-once semantics ensures that a message is delivered to the recipient at most once. If the sender fails to receive an acknowledgment from the receiver within a specified time period, it assumes that the message was not successfully delivered and retransmits it. This guarantees that the recipient does not receive duplicate messages, but it also introduces the possibility of message loss if the retransmission fails.

2. Exactly-once

The exactly-once semantics ensures that a message is delivered to the recipient exactly once without any loss or duplication. To achieve this, the sender and receiver maintain a sequence number for each message to ensure that each message is delivered in the correct order and without repetition. Additionally, the protocol includes mechanisms for detecting duplicate messages and discarding them.

3. At-least-once

The at-least-once semantics ensures that a message is delivered to the recipient at least once, guaranteeing message delivery but allowing duplicates. The sender keeps track of sent messages and if an acknowledgment is not received within a specified time period, it retransmits the message. The recipient must handle duplicate messages and ensure that any duplicates are not processed multiple times.

5.Explain how File Tracking is achieved in non-Distributed file systems [3 Marks]

File tracking in non-distributed file systems is achieved through various mechanisms, such as file metadata, file system journaling, and file locking.

1. File metadata: In non-distributed file systems, each file is assigned metadata that contains information about the file, including its name, location, size, permissions, and timestamps. This metadata is stored in a data structure called an Inode (Index Node) in most Unix-based file systems. By maintaining this metadata, the file system can keep track of the files and their properties.
2. File system journaling: Non-distributed file systems often employ journaling techniques to track file changes and maintain data consistency. In journaling, before any changes are made to a file, a copy of the change is written to a journal. This journal acts as a temporary log of all file system modifications or operations, including file creations, deletions, renames, and metadata updates. In case of system failure or power loss, the file system can replay the journal to bring it back to a consistent state.
3. File locking: In non-distributed file systems, file locking can be used to track the access and modifications to files by multiple users or processes. File locking allows exclusive or shared access to a file, preventing other users or processes from modifying it simultaneously. When a process wants to access a file, it requests a lock on that file. If the file is already locked, the

requesting process waits until the lock is released. This ensures that only one process can modify a file at a time, preventing data corruption or conflicts.

6.State the four generalized types of failures that can occur when an RPC call is made. [4 MKS]

1. Communication failures: These failures occur due to network issues such as connection timeouts, network errors, or server unavailability. The client cannot establish a connection to the server or experiences a disruption in communication.
2. Server failures: These failures happen when the server encounters an error or crashes during the execution of the RPC call. It may be due to software bugs, hardware failures, or resource limitations. In this case, the client may not receive a response or may receive an error message.
3. Parameter failures: These failures occur when the client provides incorrect or invalid input parameters to the RPC call. It may include passing wrong data types, out-of-range values, or missing required parameters. Consequently, the server may reject the call and return an error.
4. Application failures: These failures occur within the application itself, independent of communication or server issues. It could be due to logical errors, software glitches, or incorrect implementation of the RPC procedure. The client may receive unexpected or erroneous results from the server in this case.