Institute Of Technology Gopeshwar

Assignment Work – Distributed Systems

Submitted To –:
Mr. Varun Prabhakar

Submitted By –:
Preety Chauhan
CSE 4th Year
67134010100 5

ASSIGNMENT-3

Answer 1) Mutual Exclusion

Mutual exclusion is a concurrent access to a shared resource by several uncoordinated over-request is serialized to secure the integrity of shared resource. It requires that the action performed by a user on a shared resource must be atomic.

The primary objective of a mutual exclusion algo is to maintain mutual exclusion; that is to guarantee only one request accesses the CS at a time. Following characte-ristics are considered important in a mutual exclusion algorithm.

① Freedom from deadlocks

2 or more sites should not endlessly wait indefin-itely to execute CS while other sites are repeatedly executing CS. That is, every requesting site should get an opportunity to execute CS in a finite time.

② fairness

fairness dictates that requests must be executed in order they are made. Since a physical global clocks does not exist, time is determined by logical clocks. Note that fairness implies freedom from starvation, but not vice-versa.

③ Fault Tolerance

A mutual exclusion algorithm is a fault-tolerant if in the wake of a failure, it can reorganize itself so that it continues to function without any prolonged disruptions.

* To measure the performance of mutual exclusion algorithm 4 matrices are used —:

① No. of messages necessary per CS invocation

The synchronization delay, that is time required to after a site leaves the CS & before the next site enters the CS.

Response time, which is the time interval a request waits for its CS execution to be over after its request messages have been sent out.

The system throughput, rate at which the system executes requests for the CS.

System Throughput = $1/(sd + E)$.

Answer 3) ACID Properties

ACID (Atomicity, Consistency, Isolation, Durability) is a set of properties that guarantee database transactions are processed reliably.

① ATOMICITY

Atomicity requires that database modifications must follow an "all or nothing" rule. Each transaction is said to be atomic. If one part of the transaction fails, the entire transaction fails & the database state is left unchanged. It is critical that the database management system maintains the atomic nature of transaction inspite of any applications, DBMS, O.S or hardware failure.

An atomic transfer cannot be subdivided & must be processed in its entirety or not at all. Atomicity means that users do not have to worry about the effect of incomplete transactions.

② CONSISTENCY

The consistency property ensures that any transaction the database performs will take it from consistent state to another.

Consistency states that only valid data will be written to the database.

The consistency rule applies only to integrity rules that are within its scope. Thus, if a DBMS allows fields of a record to act as references to another record, then consistency implies the DBMS must enforce referential integrity

If a transaction consisted of an attempts to delete a record referenced by another, each of the following mechanism would maintain consistency.

→ abort the transaction, rolling back to the consistent, prior state

→ delete all records that referenced the deleted record

→ nullify the relevant fields in all records that point to the deleted record.

(3) ISOLATION

Isolation refers to the requirement that other operation can't access data that has been modified during a transaction that has not yet completed. If the isolation system does not exist, then the data could be put into an inconsistent state. This could happen, if one transaction is in the process of modifying data but has not yet completed, & then a second transaction reads & modifies that uncommitted data from the first transaction. If 1st transaction fails & the second one succeeds, that violation of transactional isolation will cause data inconsistency.

(4) DURABILITY

Durability is the ability of the DBMS to recover

the committed transaction updates against any kind of system failure. Durability is the DBMS's gurantees that once the user has been notified of a transaction's success. the transaction will not be lost, the transaction's data changes will survive system failure, & that all integrity constraints have been satisfied, so the DBMS won't need to reverse the transaction. Many DBMSs implement durability by writing transaction into a transaction log that can be reprocessed to recreate the system state right before any later failure. Durability does not imply a permanent state of the database. A subsequent transaction may modify data changed by a prior transaction without violating the durability principle.

Answer 2) Atomic Transaction.

Atomic transaction simply means that either the entire transaction takes place at once or doesn't happen at all. There is no midway i.e. transactions do not occur partially. Each transaction is considered as one unit & either runs to completion or is not executed at all. It involves 2 operations —:

— Abort → Changes made to database are not visible.

— Commit → If a transaction commits, changes are visible.

It's basically "all or nothing rule".

Before X: 500 Y: 200

Transaction T

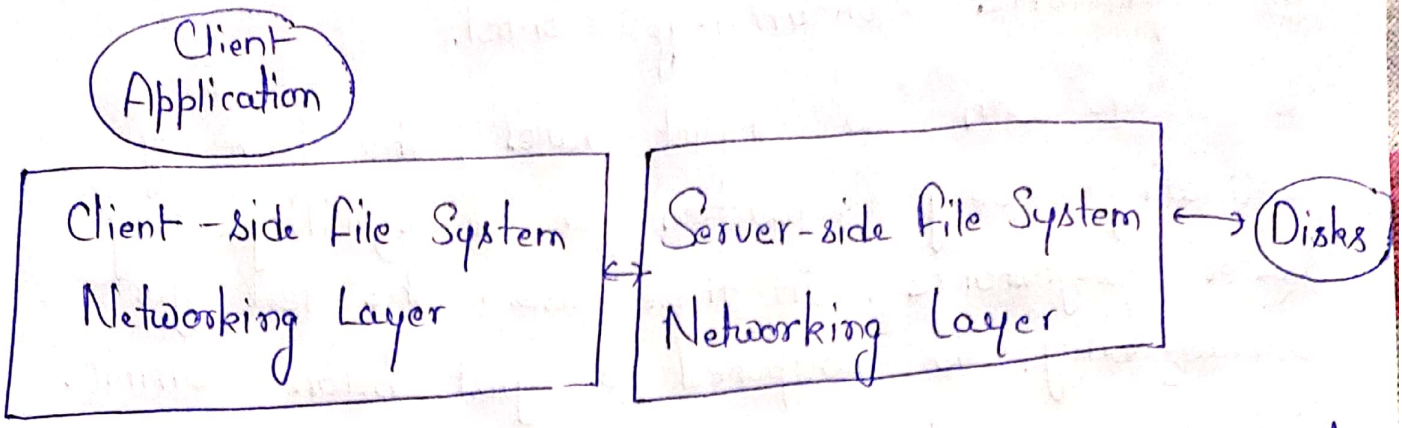
T_1	T_2
Read (X)	Read (Y)
X := X − 100	Y := Y + 100
Write (X)	Write (Y)
After: X: 400	Y: 300

If Transaction fails after completion of T_2, then amount has been deducted from X but not added to Y. This results in an inconsistent database state. Therefore, the transaction must be executed in entirety in order to ensure correctness of database state.

Answer 4)

(a) Network File System.

The advent of distributed computing was marked by the introduction of file systems. Such system invdved multiple client machines & one or a few servers. The server stores the data on its disks & the client may request data through some protocol messages.



NFS v2 developed by SUN was the standard protocol followed for many years, designed with the goal of simple & fast server crash recovery. This goal is of utmost importance in multi-client & single-server based network architectures because a single instant of server crash means that all clients are unserviced. The entire system goes down.

* File handles → NFS uses file handles to uniquely identify a file or a directory that the current operation is being performed upon.
  - (a) Volume Identifier → Tells the server which file system is being referred to.
  - (b) Inode Number → Identifies the file within the partition.
  - (c) Generation Number → This number is used while reusing an inode number.
* File Attributes → It is a collective term for the tracked metadata of a file.

(b) Optimistic Concurrency Control.

In systems with low conflict rates, the task of validating every transaction for seriazability may lower performance. In these cases, the test for serializability is postponed to just before commit. Since the conflict rate is low, the probability of aborting transactions which are not seriazable is also low. This is called Optimistic Concurrency Control.

In this phase a transaction's life cycle is divided into the following 3 phases —:

→ Execution Phase

A transaction fetches data items to memory & performs operation upon them.

→ Validation Phase

A transaction performs checks to ensure that committing its changes to the database passes serializability test.

→ Commit Phase

A transaction writes back modified data items in memory to the disk.

(c) Distributed Deadlocks.

Distributed deadlocks can neither be prevented nor avoided as the system is so vast that it is important impossible to do so. Therefore, only deadlock detection can be implemented. Techniques of deadlock detection are -:.

Progress — The method should be able to detect all the deadlocks in the system.

Safety — The method should not detect false

3 approaches to detect deadlock in distributed system.

(1) Centralized Approach

There is only one responsible resource to detect deadlock. The advantage of this approach is that it is simple & easy to implement, while the drawback include excessive workload at one node.

(2) Distributed Approach

Different nodes work together to detect deadlocks. No single point failure as the workload is equally divided among all nodes.

(3) Hierarchical Approach

It is the combination of both centralized & distributed approaches of deadlock detection in a distributed system.

(d) Domain Name System.

DNS help to resolve the host name to an address.
It uses a hierarchical naming scheme & distributed database of IP addresses & associated name.
It is a symbolic string associated with an IP address
Some Generic Top-level Domain names are -:

Com - Commercial business
Edu - Education
Gov - U.S. government agency
Int - Internation Entity.
Mil - U.S. Military.

* Name Server → Name server contain the Database. It comprises for various name & their IP addresses.

* Zones → Zone is collection of nodes under the main domain.

* Servers that manage the entire DNS -:

(i) Root Server
It is the top level server which consists of the entire DNS tree It don't contain information about domains but delegate the authority to other server.

(ii) Primary Server
Store file about its zone.

(iii) Secondary Server
It transfer complete information about a zone from another server which may be primary or secondary. server.