UNIT- 2

Date ____/____/____
Page _____

DISTRIBUTED SHARE MEMORY & DISTRIBUTED FILE SYSTEM

① Basic concept of Distributed Share Memory (DSM) -

DSM system is a resource management component of a distributed operating system that implements the shared memory model in distributed system, which have no physical shared memory. The shared memory model provides a virtual address space that is shared among all nodes in a distributed system.

DSM is also refered as Distributed Shared Virtual Memory (DSVM).

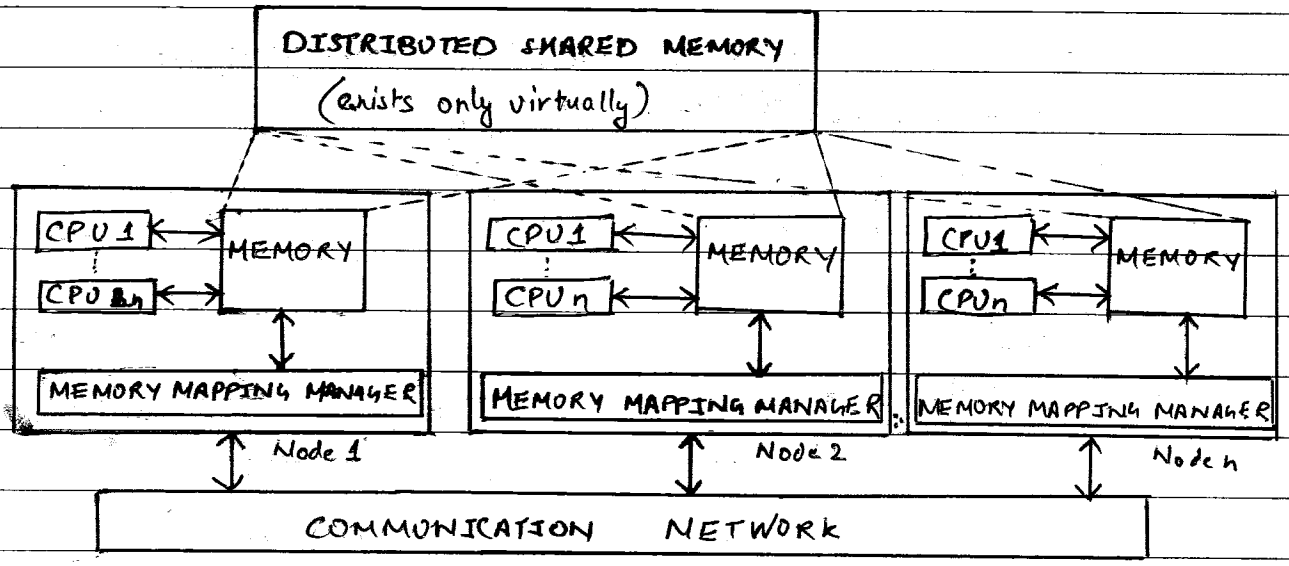
Advantages -

(1) Shields programmer from send/receive primitives
(2) Large virtual memory space.
(3) Single address space
(4) Simple software interfaces.
(5) Programs portable

Disadvantages -

(1) May incur a performance penality
(2) No protection against shared data.

② DSM Architecture and its types -



DSM ARCHITECTURE

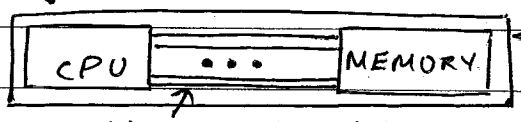
Each node A simple message-passing system allows processes on different nodes to exchange messages with each other.

A software memory-mapping manager routine in each node maps the local memory onto the shared virtual memory. To facilitate the mapping operation, the shared-memory space is partitioned into blocks.

Data caching is used in DSM systems to reduce network latency.

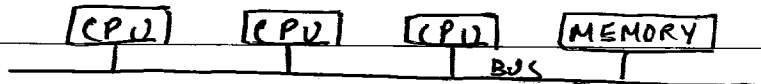
→ Types of DSM -

(1) On-chip memory - Several processors & a shared memory are on the same chip.



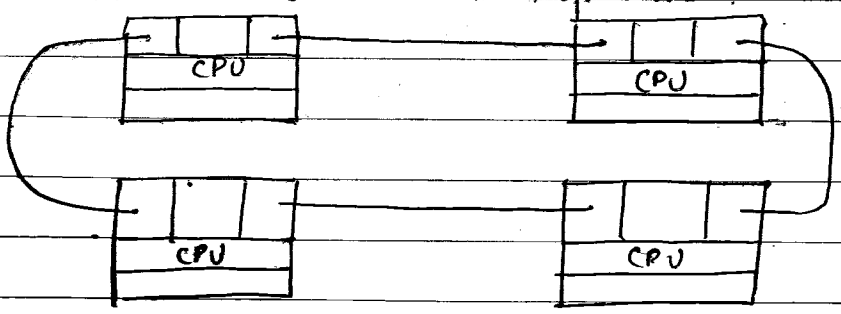
SINGLE CHIP COMPUTER

(2) Bus-based systems - CPUs and memories are connected to a BUS. CPUs either have or do not have local memories.



MULTI PROCESSOR

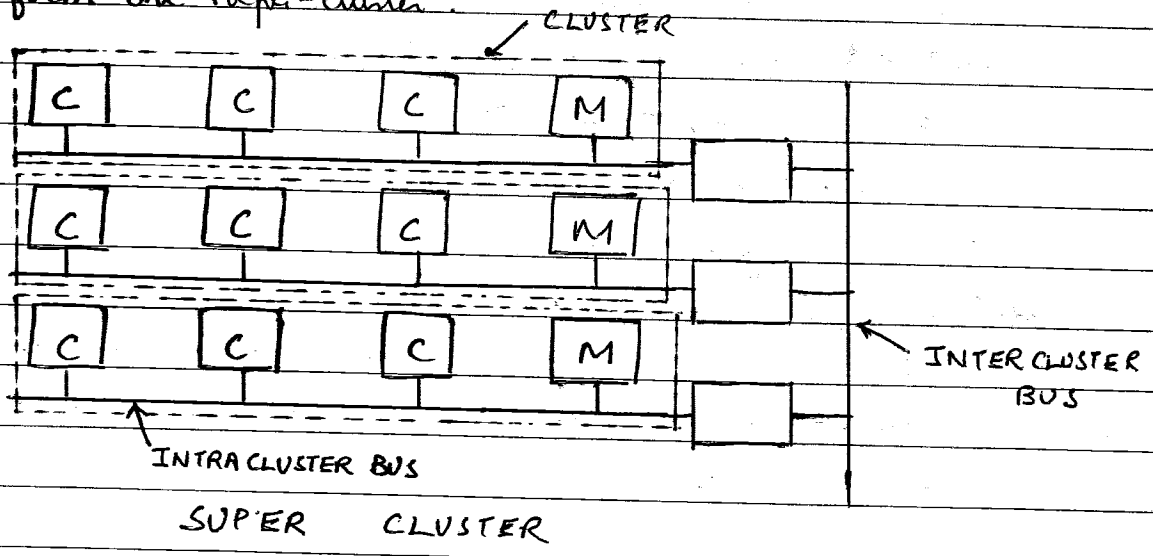
(3) Ring-based multiprocessors - Single address space is divided into a private part and a ~~public~~ shared part. The private part is divided up into regions so that each machine has a piece for its stacks and other unshared data and code. The shared part is common to all machines and is kept consistent by a hardware protocol roughly similar to those used on bus-based multiprocessors.



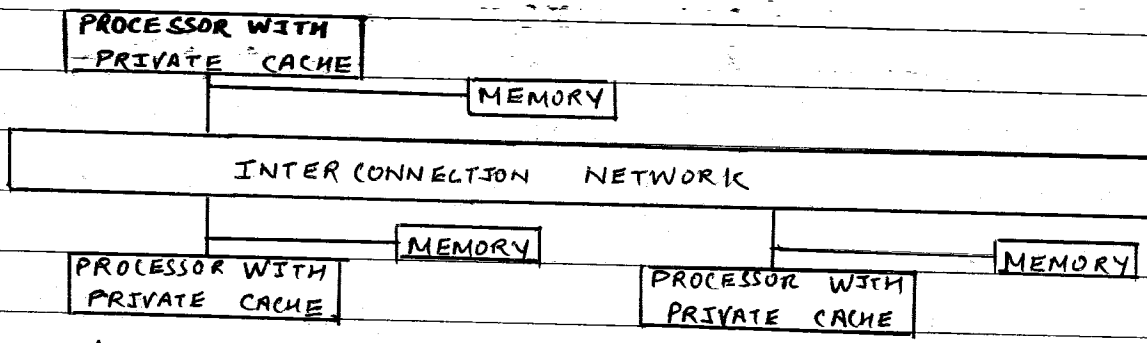
Ring based multiprocessor (Memnet Ring)

Memnet ring is a modified token-passing ring which has 20 parallel wires (16 data bits + 4 control bits) to sent bits at every 100 nsec for a data rate of 160 MBps. Shared memory is divided into 32-byte blocks

(4) Switched Multiprocessor - Three clusters connected by an inter-cluster bus to form one super-cluster.



(5) NUMA (Non-Uniform Memory Access) multiprocessors -



A processor's own internal computations can be done in its own local memory leading to reduced memory contention.

Accessing remote memory is possible but slower than accessing local memory.

Remote access times are not hidden by caching.

(3) Design and Implementation Issues in DSM System -

(1) Granularity - When a nonlocal memory word is referenced, a chunk of memory containing the word is fetched from its current location and put on the machine making the reference. An important design issue is how big should the chunk should be? A word, block, page, or segment (multiple pages).

(2) Structured of shared memory space - depends on the type of application that the DSM system intended to support.
(3) Data location and access - to share data in a DSM system
(4) Replacement strategy - Data block of local memory must be replaced by a new data block.
(5) Thrashing - Data blocks migrate between nodes on demand

→ Challenges in DSM -
(1) How to keep track of the location of remote data?
(2) How to overcome the communication delays & high overhead associated with the references to remote data?
(3) How to allow "controlled" concurrent access to shared data?

(4) Structure of shared memory space -
Three commonly used approaches for structuring -
(1) No structuring - linear array of words, simple & easy to design. Eg - IVY
(2) Structuring by data-type - Collection of objects or variables in a source language
(3) Structuring as a database
→ Shared memory space is ordered as an associative memory, called a tuple space, which is a collection of tuples with data items in their fields.

(5) Consistency Model -
It determines when the data updates are propagated and what level of inconsistency is acceptable. Models are as follows -
(1) Strict Consistency -
A shared memory system is said to support the strict consistency model if the value returned by the read operation on a memory address is always the same as the value written by the most recent write operation to that address.
Not suitable in distributed systems. Difficult to achieve in real systems as network delays can be variable.

(2) Sequential Consistency - (Strongest memory model for DSM)

A DSM system is said to be sequentially consistent if for any execution there is some interleaving of the series of operations issued by all the processes that satisfies the following two criteria -

(i) SC1 - The interleaved sequence of operations is such that if occurs in the sequence, then either the last write operation that occurs before it in the interleaved sequence is, or no write operation occurs before it.

(ii) SC2 - The order of operations in the interleaving is consistent with the program order in which each individual client executed them.

It provides single-copy semantics because all processes sharing a memory location always see exactly the same contents stored in it.

(3) Linearizability -

The result of any execution is the same as if the operations by all processes on the data were executed in some total order.

→ Bring in servers view to define the ordering of concurrent events

→ Real time actions performed on the servers.

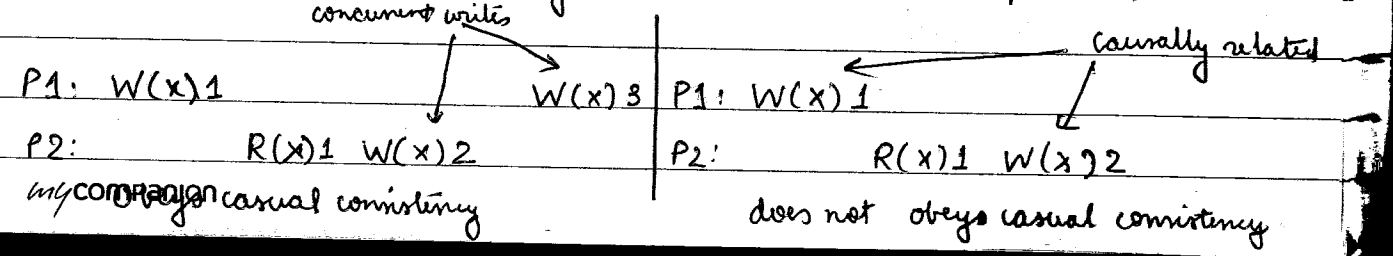
→ Non-overlapping request.

→ Overlapping request - Enqueuing times of the requests are in different orders on different servers can have arbitary order, but sequentially consistent.

(4) Causal consistency -

Operations that are causally related must be seen by all processes in the same corresponding order. Concurrent writes from different processes do not have any causal relationship and can be seen in different order by different processors. There is no need to write exclusively, cheaper write operation.

Need to keep track of dependency relations in order to determine whether two events are causally related. This model is expensive.



(5) Pipelined RAM Consistency –

It can be implemented by simply sequencing the write operations performed at each node ~~to~~ independently of the write operations performed on other nodes.

It is simple and easy to implement and also has good performance.

(6) Weak Consistency –

It has three properties –

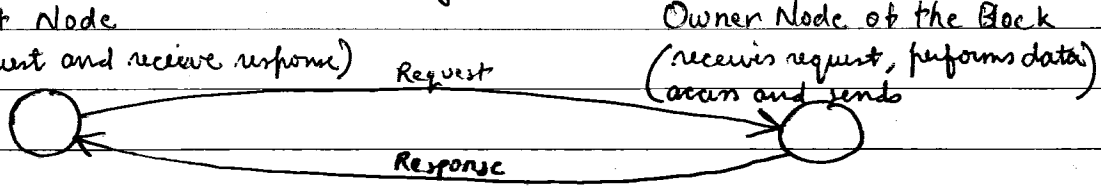
(1) Accesses to synchronization variable are sequentially consistent.
(2) No access to a synchronization variable is allowed to be performed until all previous writes have completed everywhere.
(3) No data access (write or read) is allowed to be performed until all previous accesses to synchronization variables have been performed.

Weak consistency requires the programmer to use locks to ensure reads and writes are done in the proper order for data that needs it.

→ Implementing sequential consistency model –

There are different types of replication & migration techniques –

(1) Non Replicated, Non Migrating Blocks (NRNMBs)



Characteristics –

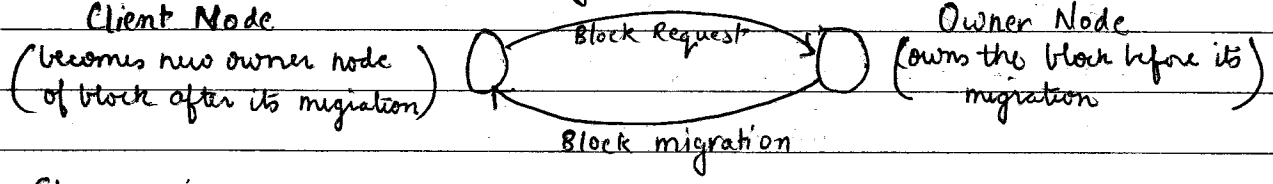
(1) Single copy of each ~~Block~~ block in the entire system.
(2) Location of a block never changes.

Drawbacks –

(1) Serializing data access creates a bottleneck.
(2) Parallelism is not possible.

This method is simple and easy to implement.

(2) Non Replicated, Migrating Blocks (NRMBs) -



Characteristics -

(i) No communication cost incurred when a process access data currently held locally.

(ii) Allow applications to take advantage of data access locally.

Drawbacks -

(i) Prone to thrashing problem (poor performance)

(ii) Parallelism is not possible.

(3) Replicated, Migrating Blocks (RMBs) -

Characteristics -

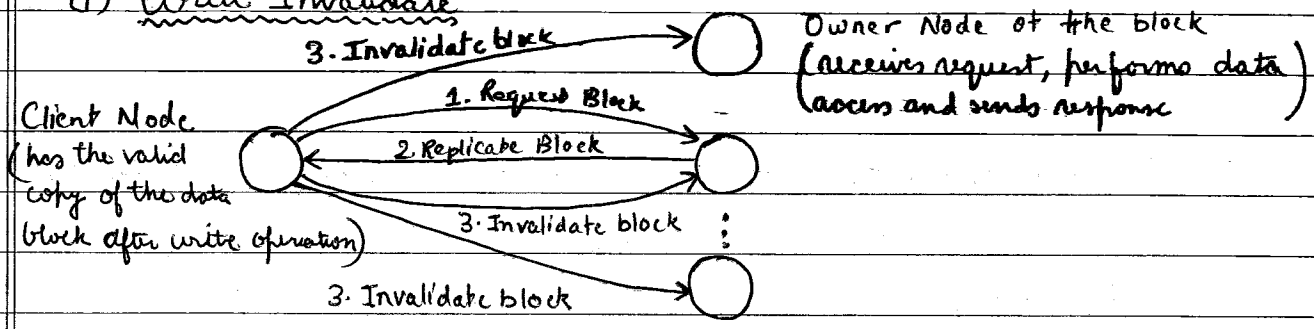
(i) Parallelism is possible due to application of Blocks.

Drawbacks -

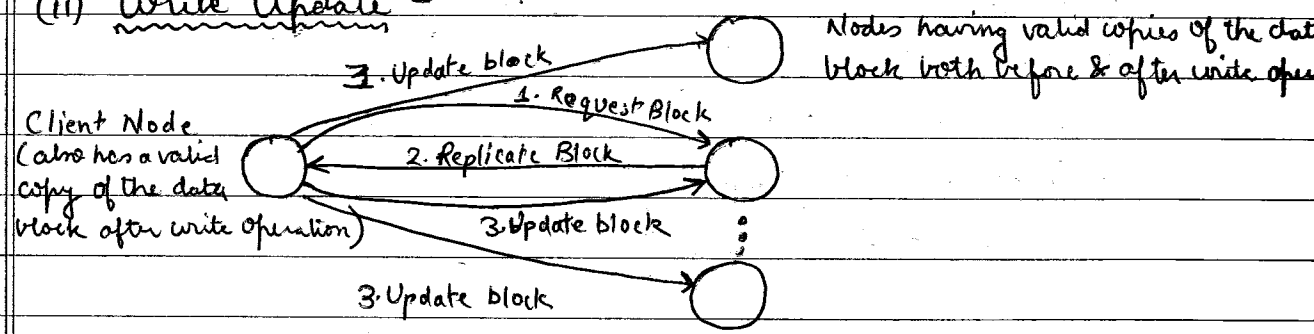
(i) Increases the cost of write operation to be perform in all replicas.

→ Two protocols is used ~~to~~ for ensuring sequential consistency -

(i) Write Invalidate -



(ii) Write Update -



(4) Replicated Non-migrating Blocks (RNMBs) -

→ Replication of blocks.
→ location of Replica is fixed.
→ Write-update protocol is used.
→ Sequential consistency is ensured by using a Global-sequencer.

(6) Thrashing -

It occurs when network resources are exhausted, and more time is spend invalidating data and sending updates that is used doing actual work.

Two or more process try to write the same shared block.

The larger the a block, the more chances of false sharing (i.e. it occurs when two different process access two unrelated variables that reside in the same data block) that causes thrashing.

Solutions -

(1) Allow a process to prevent a block from accessed from the others, using a lock.
(2) Allow a process to hold a block for a certain amount of time.
(3) Apply a different coherence algorithm to each block.

DISTRIBUTED FILE SYSTEM -

(1) Distributed file system is a resource management component of a distributed operating system. It provides a user with a unified view of the files on the network. A machine that holds the shared files is called a server. A machine that accesses the files is called a client.

Goals of distributed file systems are -
(1) Network transparency
(2) High availability.

Date ___/___/___
Page _____

(2) Desirable features of good distributed file system -

(1) Transparency -

(i) Structure Transparency - Client should not know the number or locations of the file servers and storage devices.

(ii) Access Transparency - Both local and remote file should be accessible in the same way.

(iii) Naming Transparency - Name of the file should give no hint as to where the file is located.

(iv) Replication Transparency - Clients do not need to know the existence or locations of multiple file copies.

(2) User mobility -

User should not force to work on a specific node but should have the flexibility to work on different nodes at different times.

(3) Performance -

Average amount of time needed to satisfy client request

(4) Scalability

(5) High availability

(6) High reliability

(7) Security.

(3) File models -

File model are based on following criteria -

(1) Unstructured and structured files -

Simple file model - File is an unstructured sequence of data. There is no substructure known to the file server. Eg - UNIX, MSDOS

Structured file model - A file appears to the file server as an ordered sequence of records. They are two types -

Indexed records - Records have one or more key fields and can be addressed by specifying the values of the key fields. Eg - B-tree, RSS, Oracle.

Non-Indexed records - A file record is accessed by specifying its position within the file. Eg - IBM mainframe

Date __/__/__
Page _____

*****

Most of the modern operating system uses the unstructured file model. This is mainly because sharing of a file by different applications is easier with the unstructured file model as compared to the structured file model.

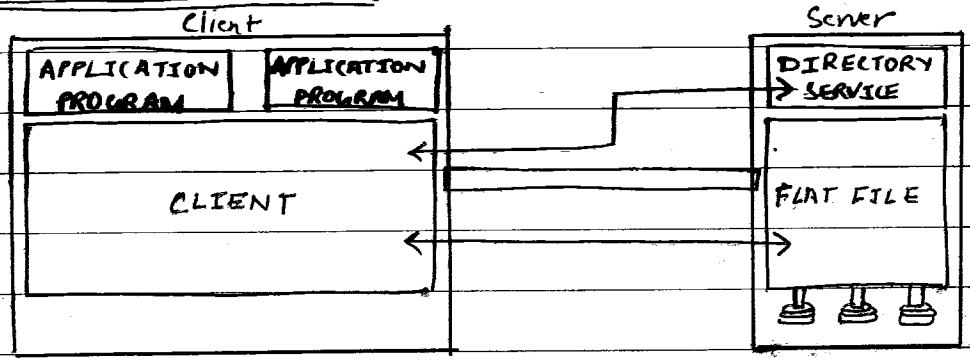
(2) Mutable and Immutable files -

Mutable files - An update is performed on a file overwrites on its own old contents to produce the new contents. File is represented as a single stored sequence that is altered by each update operation.

Immutable files - A file cannot be modified once it has been created except to be deleted. File versioning approach is used. Eg - Cedar file system

Problems - Increased use of disk space and disk allocation activity.

(4) File Service Architecture -



There are three components -

(1) Flat file Service -

Concerned with the implementation of operations on the contents of file. Unique File Identifiers (UFIDs) are used to refer to files in all requests for flat file service operations.

(2) Directory Service -

It provides mapping between text names for the files and their UFIDs. Client may obtain the UFID of a file by quoting its text name to directory service.

(3) Client module -

It provides integrated service (flat file & directory) as a single API to application programs. It holds information about the network

location of flat-file and directory servers processes; and achieves better performance through implementation of a cache of recently used file blocks at the client.

→ Flat file service operations -

(1) Read (FileId, i, n) → Data - throws BadPosition -

If $1 \leq i \leq$ Length(File): Reads a sequence of up to n items from a file starting at item i and returns it in Data.

(2) Write (FileId, i, Data) - throws BadPosition -

If $1 \leq i \leq$ Length(File)+1: Write a sequence of Data to a file, starting at item i, extending the file is necessary.

(3) Create () → FileId -

Creates a new file of length 0 and delivers a UFID for it.

(4) Delete (FileId) - Removes the file from the file store.

(5) GetAttributes (FileId) → Attr - Returns the file attributes for the file.

(6) SetAttributes (FileId, Attr) - Set the file attributes.

→ Directory service operations -

(1) Lookup (Dir, Name) → FileId - throws NotFound -

locates the text name in the directory and returns the relevant UFID.

(2) AddName (Dir, Name, FileId) - throws NameDuplicate -

If Name is not in the directory, adds (Name, File) to the directory and updates the file's attribute record.

(3) UnName (Dir, Name) - throws NotFound -

If Name is in the directory, removes the entry containing Name from directory.

(4) GetNames (Dir, Pattern) → NameSeq -

Returns all the text names in the directory that match the regular expression ~~pattern~~ Pattern.

→ UNIX file system uses access control, hierarchy file system and file group (it is a collection of files that can be located on any server or moved between servers while maintaining the same names).

⑤ File Accessing Models –

It depends on two factors –

(1) The method used for accessing remote files (2) The Unit of data access.

→ Accessing Remote files – Two models –

(1) Remote Service model –

Clients request for the file access is delivered to the server, the server machine performs the access request, and finally the result is forwarded back to the client. Request are transferred as messages.

Merit – A simple implementation Demerit – Communication Overhead

(2) Data Caching Model –

If the data needed to satisfy the client's access request is not present locally, it is copied from the server's node to the client node and is cached there.

Merit – Reducing network traffic Demerit – Cache consistency problem.

→ Unit of Data Transfer – Four models –

(1) File level transfer model – Complete file is moved. Eg – AFS, Amoeba

Merits – Simple, less communication overhead and immune to server

Demerits – A client required to have large storage space. Eg –

(2) Block level transfer model – Units of file blocks are moved Eg – NFS

Merits – A client not required to have large storage space.

Demerits – More network traffic / overhead

(3) Byte level transfer model – Units of bytes are moved. Eg – Cambridge file server

Merits – Flexibility maximized

Demerits – Difficult cache management to handle the variable-length data

(4) Record level transfer model – Units of records are moved.

Merits – Handling structured and indexed files

Demerits – More network traffic and more overhead to re-construct of a file

Date ___/___/___
Page _____

⑥ File Sharing Semantics -

Define when modifications of the file data made by a user are observable by other users.

(1) UNIX Semantics -

A file is associated with a single physical image that is associated as an exclusive resource. Contention for this single image causes delays in user process. It is used in centralized and single processor systems.

Unix file system implements -

(1) Write to an open file visible immediately to other users of the same open file
(2) Sharing file pointer to allow multiple users to read and write concurrently.

(2) Session Semantics -

No changes are visible to other processes until the file is closed. A file can be associated with multiple users. Almost no constraints are imposed on scheduling accesses. No user is delayed in reading or writing their personal copy of the file. Eg - Andrew File System (AFS)

(3) Immutable Shared file semantics -

No updates are possible; simplifies sharing and replication.

(4) Transactions like semantics -

All changes occur atomically. Begin transactions, perform operations and end transaction. The final file content is the same as if all transactions were run in some sequential order.

⑦ File Caching Schemes -

It is used to improve the I/O performance by reducing disk transfers. It also addresses the following key decisions -

(i) Cache location
(ii) Modification propagation
(iii) Cache validation.

~~Cost of cache miss = cost of access disk + network transfer~~

~~Cost of cache hit = Network transfer~~

Date ___/___/___

Page _____

→ Cache location -

Cache behaves just like "networked virtual memory."

(1) Server's main memory -

Cost of cache miss = cost of access disk + network transfer

Cost of cache hit = Network transfer

Merits - One-time disk access, Easy implementation, Unix like file sharing semantics.

Demerits - Busy Network traffic.

(2) Client's disk -

Cost of cache hit = Time to access from local disk.

Merits - One-time Network Access, No size restriction, Suitable for supporting disconnected operation.

Demerits - Cache consistency problem, file access semantics, Frequent disk access, No diskless workstation.

(3) Client's Main Memory -

Cost of cache hit = maximum (high performance)

Merits - Maximum Performance, Diskless workstation, Scalability.

Demerits - Size restriction, Cache consistency problem, file access semantics

→ Modification Propagation -

The aim is keeping file data cached at multiple client node consistent. It has a critical effect on the system's performance and reliability. The file semantics supported depends greatly on the modification propagation scheme used.

Cache update policy -

(1) Write through - When a cache entry is modified, the new value is immediately sent to the server for updating the original copy of the file.

Pros - Unix like semantics and high reliability

Cons - Poor write performance.

(2) Delayed Write - The aim is to reduce network traffic for writes. When a cache entry is modified, the new value is written only to the cache.

and client just makes a note.

Pros - Write accesses complete quickly, some writes may be omitted by the following writes, Gathering all writes mitigates network overhead.

Cons - Delaying of write propagation results in fuzzier file-sharing semantics.

→ Cache Validation -

Cache consistency is another important issue that determines how caches are maintained when multiple clients may be accessing the same file.

Client-initiated approach -

Client is responsible for checking with the server to verify that each file in its cache is consistent.

A single corrupt or malicious client could disrupt the complete system.

Server-initiated approach -

Server acts as a central authority over which clients have up to date or invalid caches.

Server is able to detect when reading and writing clients might conflict with each other, and will send messages to client to force them to invalidate their cache entries and request them again.

Concurrent write sharing approach -

A file is open at multiple clients and at least one client has its open for writing. File server keep track of the clients sharing a file.

(8) File Replication -

A replicated file is a file that has multiple copies, with each file on a separate file server.

Advantages of replication -

(1) Increased Availability
(2) Increased Reliability
(3) Improved Response time
(4) Reduced Network traffic
(5) Improved system throughput
(6) Better scalability

Date __/__/__
Page _____

Differente between Replication and Caching -

(1) A replica of a file is associated with server, whereas a cached copy is normally associated with a client

(2) The existence of a cached copy is primarily dependent on the locality in file access patterns, whereas the existence of a replica normally depends on availability and performance requirements

(3) As compared to cached copy, a replica is more persistent, widely known, secure, available, complete and accurate.

(4) A cached copy is contingent upon a replica. Only by periodic revalidation with respect to a replica can a cached copy be useful.

Replication Transparency -

Replication of files should be transparent to the users so that multiple copies of a replicated file appear as a single logical file to its users. This calls for the assignment of a single identifier/name to all replicas of a file.

In addition, replication control should be transparent i.e. the number and locations of replicas of a replicated file should be hidden from the user. Thus replication control must be handled automatically in a user-transparent manner.

Multicopy Update Problem -

Major design issue of a distributed file system that supports file replication. TO ~~remove it~~ avoid this problem, we can use -

(1) Read only replication protocol.

(2) Read-any-write-all protocol.

(3) Available-copies protocol - Same as (2) protocol but writes to all available copies of the file

(4) Primary-Copy protocol - Read operations can be performed using any copy, primary or secondary but write operations are performed only on the primary copy. Each server having a secondary copy updates its copy.

Date ____/____/____
Page _____

⑨ Fault Tolerance -

The approach of fault-tolerance expect faults to be present during system operation, but employs design techniques which insure the continued correct execution of the computing process.

The primary file properties that directly influence the ability of a distributed file system to tolerate faults are as follows:

(1) Availability (2) Robustness (power to survive crashes) (3) Recoverability

Stable Storage -

Information never lost. Not actually possible, so approximated via replication or RAID to devices with independent failure modes.

Two basic operation -

(1) A read operation first attempts to read from disk1. If it fails, the read is done from disk2.

(2) A write operation writes to both disks, but the write to disk 2 does not start until that for disk1 has been successfully completed.

It is suitable for applications that require high degree of fault tolerence. Eg - Atomic transactions

Effects of Service Paradigm on fault tolerance -

The file servers that implement a distributed file service can be stateless or stateful

(1) Stateful file server -

Server maintains information about a file opened by a client that means it store session state. File operations supported by this server are -

(1) Open (filename, mode)
(2) Read (fid, n, buffer)
(3) Write (fid, n, buffer)
(4) Seek (fid, position)
(5) Close (fid) → causes the server to delete from its file table the file state information of the file identified by fid.

→ Advantages of Stateless servers -

(1) Fault Tolerance
(2) NO OPEN/CLOSE calls needed
(3) NO server space wasted on tables
(4) NO limits on number of open files
(5) NO problem if a client crashes

Advantages of Stateful servers -

(1) Shorter request messages
(2) Better performance
(3) Read ahead possible
(4) Idempotency easier
(5) File locking possible

→ Disadvantages of stateful servers -

(1) Problem of orphan detection of elimination
(2) It loses all its volatile state in a crash

Disadvantages of Stateless servers -

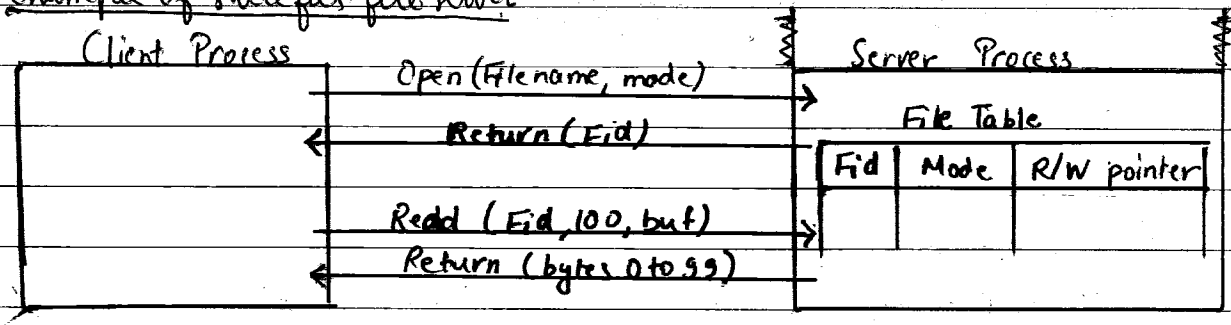
(1) less performance
(2) There is no longer request messages and slower processing of request.

(2) Stateless file server -

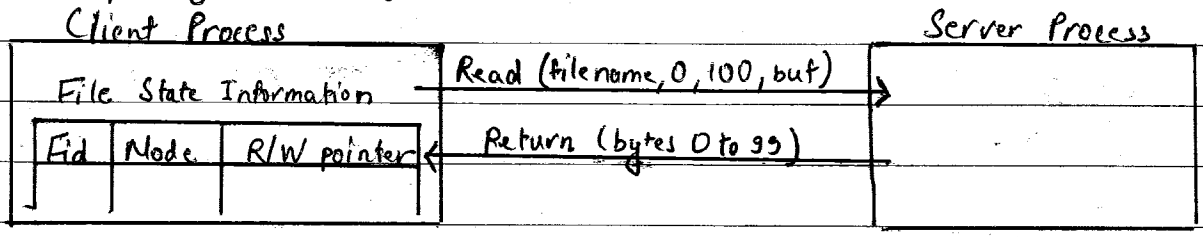
Server maintains no information about client access to files that means it do not store any session state. File operations are given as -

(1) Read (filename, position, n, buffer)
(2) Write (filename, position, n, buffer)

→ Example of stateful file server -



→ Example of stateless file server -



Date ___/___/___
Page _____

Naming –

(1) The naming facility (assign character-string names to objects) and locating facility (maps an object's name to the object's location) jointly form a naming system that provides the users with an abstraction of an object that hides the details of how and where an object is actually located in the network.

(2) Desirable features of a good naming system –

(1) Location Transparency
(2) Location Independency
(3) Scalability
(4) Uniform Naming Convention
(5) Multiple user defined names for the same object
(6) Group Naming
(7) Meaningful Names
(8) Performance
(9) Fault Tolerance
(10) Replication Transparency

(3) System Oriented Names – (low level names)

They are of fixed size bit pattern that can easily manipulate and stored by machines. It basically meant for use by the system but may also be used by the user.

Characteristics –

(1) They are larger integers or bit strings.
(2) Also referred as unique identifier
(3) length is variable
(4) Automatically generated
(5) They are hard to guess and provide good security.
(6) They are suitable for efficient handling by machines.

→ Centralized approach for generating system-oriented names –

It generates structured and unstructured names. A standard & uniform global identifier name is generated for each object in the system by a centralized global unique identifier generator.

UNSTRUCTURED NAMES

A single field of large integers or bit strings

STRUCTURED NAMES

Node Identifier	Local Unique Identifier

Date ___/___/___
Page _____

Advantages -
(1) Simple and easy to implement
(2) Only method used for generating unstructured global unique identifiers.
Disadvantages -
(1) Poor efficiency and poor reliability
(2) Single global unique identifier generator may become a bottleneck for large name space.

→ Distributed approach for generating system-oriented names -
Hierarchical concatination method is used to create global unique identifiers by concatenating the unique identifier of a domain.
Advantages -
(1) Better efficiency and reliability than centralized approach.
Disadvantages -
(1) Node boundaries or servers are explicitly visible
(2) The form and length of identifier may be different for different computer resulting in non-uniform global unique identifier.

(4) Object locating Mechanisms -
It is the process of mapping an object's system-oriented unique identifier to the replica locations of the object. Various Types are -
(1) Broadcasting - Eg - Amoeba
A request is broadcast which is processed by all nodes and then the nodes currently having the object reply back to the client node
Advantages - simple, high reliability.
Disadvantages - Poor efficiency, poor scalability, number of node should be small, needed high communication speed.
(2) Expanding Ring Broadcast -
Modified form of broadcasting method. It consists of LAN connected by gateways. The distance metric used is a hop. A hop corresponds to a gateway between processors.

A name space is a collection of names which may or may not share an identical resolution mechanism.

It supplies nearest replica location but not necessarily all replica locations.

(3) Encoding location of object within its V-ID -

It uses structured object identifiers. It is straightforward and efficient scheme.

limitations -

(1) It is not clear how to support multiple replicas of an object
(2) An object is not permitted to move once it is assigned to a node.
(3) An object is fixed to one node throughout its lifetime.

**** One solution is to use forward location pointers but it increases object-locating cost, additional system overhead and difficult to locate if an intermediate pointer has been lost.

(5) Human-Oriented Names - (High level Names)

It is generally a character string and it is meaningful and identified by its user. Not unique for an object and are normally variable in length.

They cannot be easily manipulated, stored and used by the machines for identification purpose.

Characteristics -

(1) They are defined and used by the user
(2) Different users can define and use their own suitable names for a shared object
(3) Due to the facility of aliasing, the same name may be used by two different users at the same time to refer to two different objects.