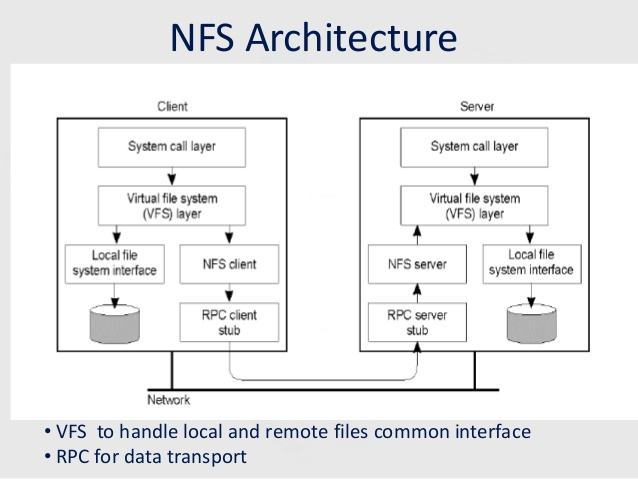
1. **What do you mean by file and directory service? Explain the operation of SUN NFS with its architecture [3+5]**  
   file service is Concerned with the implementation of operations on the contents of file. Unique File Identifiers (UFIDs) are used to refer to files in all requests for flat file service operations. UFIDs are long sequences of bits chosen so that each file has a unique among all of the files in a distributed system.

Directory Service Provides mapping between text names for the files and their UFIDs. Clients may obtain the UFID of a file by quoting its text name to directory service. Directory service supports functions needed generate directories, to add new files to directories.



**Sun’s Network File System:**

The earliest successful distributed system could be attributed to Sun Microsystems, which developed the Network File System (NFS). NFSv2 was the standard protocol followed for many years, designed with the goal of simple and fast server crash recovery. This goal is of utmost importance in multi-client and single-server based network architectures because a single instant of server crash means that all clients are unserviced. The entire system goes down.

**1. Protocol:**

- It uses SUN RPC mechanism and SUN external data representation (XDR) standard.

- The protocol is stateless. It enhances crash recovery.

- Each procedure call must contain all the information necessary to complete the call.

**2. Server Side:**

- It provides file handle consisting of:

a) Filesystem id (identify disk partition)

b) I-node number (identify file)

c) Generation number

- File system id is stored in super block.

- Generation number is stored in I-node.

**3. Client Side:**

- It provides transparent interface to NFS.

- Mapping between remote file name and remote file address is done at server boot time

through remote mount.

**operations:**

* Search for file within directory
* Read a set of directory entries
* Manipulate links and directories
* Read/write file attributes
* Read/write file data

1. **Define distributed objects. How distributed objects communicate with other? [2+6]**

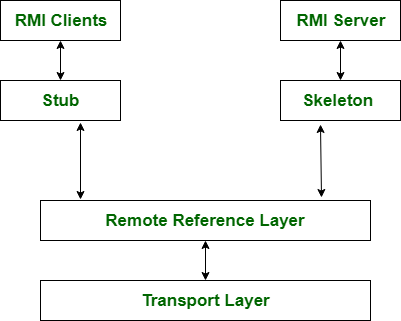
Distributed objects are the objects with respect to object oriented programming paradigm,

that are distributed across different address spaces, which may be on multiple computers

within a network or on multiple processes running on a same computer.

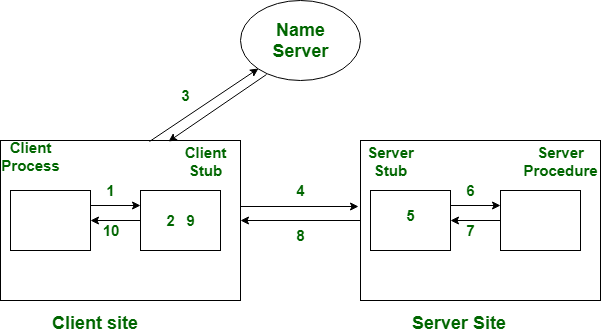
***Remote Method Invocation (RMI)***

* RMI is the means by which objects in different processes can communicate with one another.
* It allows object in one process to invoke or call the methods of an object in another process.
* Each process consists of a set of objects. These objects may receive local invocation or remote invocation or both.
* The method invocation between the objects in different processes within a computer or different computer is known as remote method invocation.
* The method invocation between the objects in same process is known as local method invocation.
* The objects which can receive remote invocations are called remote objects
* All the objects can receive local invocation from another object within a same process if and only if the invoking object have a reference to the invoked object.

****

**Remote Procedure Call**

* RPC is a remote communication medium in which a client program calls a procedure in another program running in a server process.
* The role of server and client is designated based on the work which is temporarily assigned. It means the server may also be clients of other servers creating a chains of RPC.
* A server process consists of a service interface which defines the procedures available for remote calling.
* RPC do not consists of object references.
* RPC also lacks the ability to create new instances of objects.
* It is implemented with maybe, at-least-once and at-most-once invocation semantics.
* It works under request-reply protocol omitting the object reference from request messages.



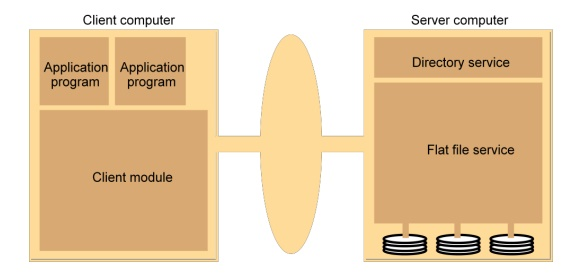
1. **Define distributed file system (DFS)? What are the different requirements of the distributed file system?[2+4]**

A Distributed File System (DFS) , is a file system that is distributed on multiple file servers or multiple locations. It allows programs to access or store isolated files as they do with the local ones, allowing programmers to access files from any network or computer.

***Requirements of DFS***

1. The design of DFS must support transparency of access, location, migration , performance, etc.
2. Concurrent file updates should be controlled.
3. It should support file replication to enhance scalability and fault tolerance.
4. The service interfaces should be defined such that they can be implemented in heterogeneous systems.
5. The service should continue to operate even in case of client or server failures.
6. It should maintain consistent states for files.
7. It should provide access control mechanisms to provide security.
8. **Define distributed file system (DFS)? How does DFS encourage sharing a storage device, explain in detail with the help of a distributed file service architecture [2+6]**

A Distributed File System (DFS) as the name suggests, is a file system that is distributed on multiple file servers or multiple locations. It allows programs to access or store isolated files as they do with the local ones, allowing programmers to access files from any network or computer.



- Distributed file system consists of three components:

a) Flat file service

b) Directory service

c) Client module

***Flat file service***

* It is a component responsible for implementing operations on the file contents.
* During requests for file service operations, Unique File Identifier (UFID) is used to indicate the files.
* The various flat file service operations are: Read, write, create, delete, getAttributes and setAttributes.

***Directory Service***

* It is a component that provides mapping between text names for files and their UFID.
* The various directory service operations are: lookup, addName, unName and getNames.

**Client module**

* It is a component running in client computer, integrating the file service and directory service operations under a single API.
* With the use of cache, client module helps to achieve high performance.

user can access the file using file identifier and also lookup the files using directory service. so no file need to be stored locally ,thus it explains the way of sharing a storage device.

1. **Define transparency in distributed system. Give the reason for “access transparency is not maintained by conventional RPC”. How can it be solved, explain with detail modern RPC procedure?[4+4]**

A transparency is some aspect of the distributed system that is hidden from the user (programmer, system developer, user or application program). A transparency is provided by including some set of mechanisms in the distributed system at a layer below the interface where the transparency is required

**Access transparency –**Both local and remote files should be accessible in the same manner. The file system should be automatically located on the accessed file and send it to the client’s side.

Access transparency can’t be solved using conventional method because remote isn't local. In particular, there are failure modes that WILL occur that do not happen in local calls, and the performance characteristics are much different even in the best cases.

modern RPC helps to solve this problem by following way:

* Library makes an API available to locally running applications
* Let servers export their local APIs to be accessible over the network, as well
* On client, procedure call generates request over network to server
* On server, called procedure executes, result returned in response to client

1. **what is stateful and stateless service in file system? Explain the DNS working with mechanism with suitable practical example [2+6]**

Stateful services is the one which keep track of sessions or transactions and react differently to the same inputs based on that history and does not require server to restrain information about the state.

Stateless services is the one which rely on clients to maintain sessions and center around operations that manipulate resources, rather than the state and require serve to restrain information about the state.

DNS is the Internet naming scheme that allows resources to be accessed by using alphanumeric names.

working mechanism of DNS:

when we wish to open the website google.com following process take place.

1. Issue a DNS query to ask for IP address of www.google.com

2. Issue a query to root name server.

3. Returns the IP address of TLD.

4. Issue a query and sent to TLD servers.

5. Reply with ns1.google.com and IP address.

6. Issue another query and sent to ns1.google.com

7. Reply IP address of www.google.com

8. Return IP address of www.google.com to client.

9. Request for web content to IP address of [www.google.com](http://www.google.com)