1.4 Issues in Designing Distributed Systems

Issues in designing distributed systems are :

- Transparency
- Flexibility
- Reliability
- Performance
- Scalability
- Security
- Fault tolerance

1.4.1 Transparency :

Processes and resources are physically distributed across network and this information is to be hidden from users. Various types of transparencies are:

Access transparency:
Machines are heterogeneous across the network i.e machines use different data representations and they operate on different platforms. This heterogeneous must be hidden from user and applications
Location transparency:
Location transparency means any user able to use resources irrespective of location. To achieve this logical names is assigned to resources,which does not depend on physical location.
Migration transparency:
This means that even though the location of resource is changed its name should not be changed and should not change how it is accessed. Ex:
If an process is transferred from one node to another node ,resource should be available to access.

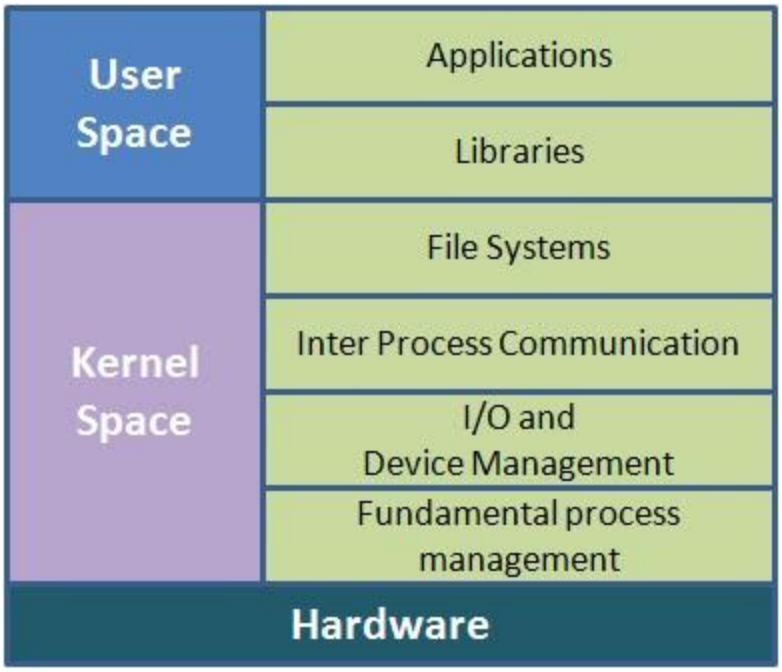
Relocation transparency :
When a process is moved from one node to another node while its in use, other processes have no idea about the movement.
Replication transparency:
This improves the availability of resources and performance, whenever resource is requested near by resource will be given. We can implement replication transparency by naming all replica the same name.
Concurrency transparency:
In distributed systems resources are shared by many systems hence it is necessary to maintain concurrency between users accessing resources. Concurrency transparency is achieved if users are unaware of each other. Resources are locked when someone is using resource. And is unlocked when user completes the task.
Failure transparency:
Errors and recoveries of nodes are invisible to users and application. If user cannot understand about failure of nodes then such system are called failure transparency. If any server is down then user must be immediately redirected to the active server. Problem is to identify whether the node is really dead or it is slow.
Persistence transparency:
This means that user should not be aware of whether the resource is in volatile memory or disk. Usually resources are brought to memory and modified resources are put into the disks. But user is unaware of it.

1.4.2 Flexibility

In initial stages modifications are done in distributed systems. So it is necessary to choose between monolithic kernel or microkernel. When process executes in supervisory mode it has access to physical resources. But when it is in the user mode it has restricted access.

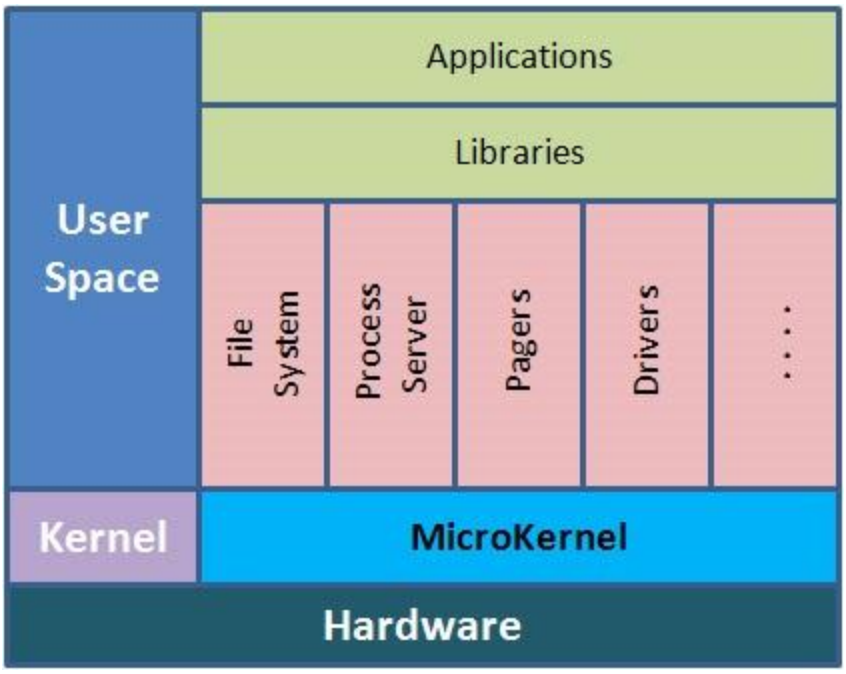
Monolithic kernel approach:

In monolithic kernel kernel does provide all the facilities. All system function are executing in the kernel space. User space has only library. Monolithic kernels are massive and non-modular in structure.functions required to carry out process management , device management, interprocess communication are done in kernel space.



Microkernel;

This uses minimalistic approach and accepts only important functions. And other functions such as process server to manage interprocess communications are available on request.microkernel are modular and small. Services like file, directory and process management are done in the user space.each services has the well-defined interface.



1.4.3 Reliabilility:

When one node crashes another node should be available to give service to user. Data should be available without any errors and replicas must be consistent. Hence multiprocessor is better than uniprocessor.

1.4.4 Performance

A process should run in distributed system similar to as it runs on single processor system. Metrics to measure the performance are throughput, response time,system utilization, amount of network capacity is used. There are two approaches to improve the performance, one is to batch the messages and send, which will reduce the traffic and another is cache the data that is needed repeatedly.

1.4.5 Scalability

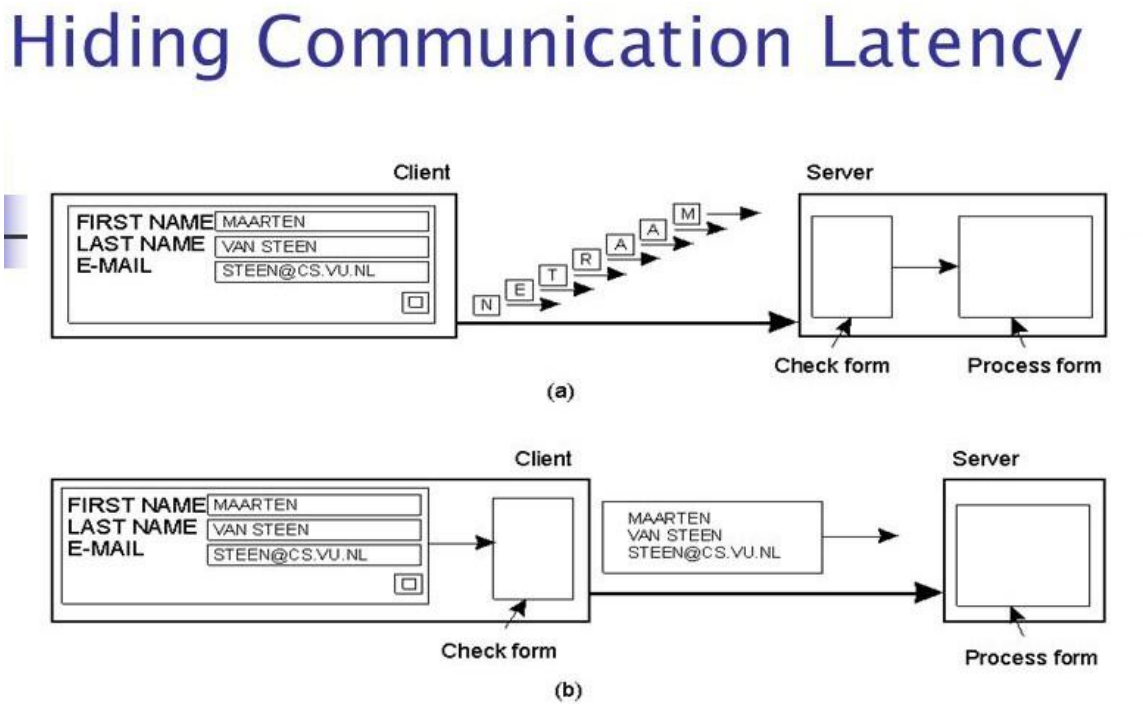
Whenever the user increases servers must also increase to give better service. The problem in the centralised services is , it has single processor which cannot handle many request at a time. Hence we need to distribute responsibility among multiple processor.

Three techniques used for scaling are :

- Hide communication latencies
- Hide distribution
- Hide replication

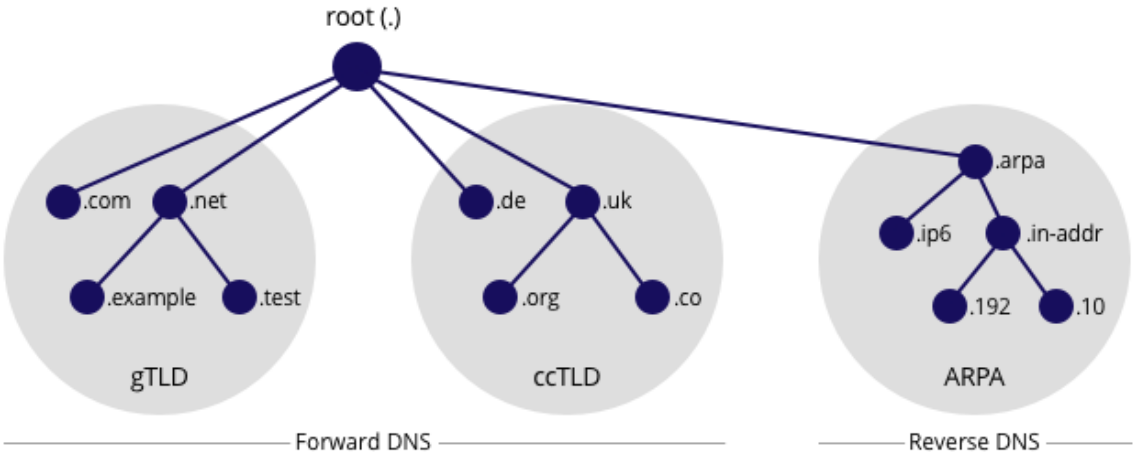
Hide communication latencies:

Whenever the servers are busy , user can use that time to do some useful work.and when reply is received application is interrupted and reply is accepted. And when user don't have anything to do , then user will be shared with the servers load.



Hide distribution:

In this approach component is divided into several component and shared across the servers. For example , DNS, where identifying task is distributed to all servers not only to single server.



Here each server has name and identifying host does not depend upon single server. Ex : .example.net. Is the name of host “example”

Hide replication :
In this approach components are replicated hence increases the availability of resources. One disadvantage is consistency problem.

1.4.6 security

Security is the most important aspect in distributed system. The main aspects of security are confidentiality-protecting data against unauthorised access ,integrity-protecting data from corruption and availability-maintaining data and always being accessible.

1.4.7 fault tolerance

Whenever the server crashes other server takes the load of it. This process is hidden from user. This allows the system to handle with crashes or any failures.