Introduction to Distributed Systems

- Software concepts of distributed Systems
- Objectives
- Types of distributed systems

A distributed system is a collection of independent computers that appears to its users as a single coherent system.
Distributed Systems

Hardware concepts – processor and memory organization
- Multiprocessor – with shared memory
- Multicomputers – without shared memory
  - Homogeneous Multicomputer System - Essentially only a single interconnection network where all processors are the same
  - Heterogeneous Multicomputer System - May contain a variety of different, independent computers, connected through different networks

Software concepts
- It is mainly the software that determines the shape of a distributed system
  - DOS, NOS, Middleware
Distributed Operating System

- Same as traditional operating system for uniprocessor system except that a DOS handles multiple CPUs
- Types of DOSs
  - Multiprocessor operating system
  - Multicomputer operating system
  - Distributed shared memory system
Network Operating System

- A collection of uniprocessor systems, each with its own operating system
- The underlying hardware is not homogenous but managed as if it were a single system
Middleware - Motivation

- A distributed system is a collection of independent computers that appears to its users as a single coherent system.
- DOS – not intended to handle a collection of independent computers.
- NOS – does not provide a view of a single coherent system.
- To have the best of both worlds – an additional layer of software in NOS: Middleware.
Distributed System as a Middleware

- **Middleware** – a layer of software logically placed between users and applications, and operating systems.

- A distributed system organized as middleware to support heterogeneous computer networks while appearing as single system.

- Note that the middleware layer extends over multiple machines.
The protocols used by each middleware layer should be the same, as well as the interfaces they offer to applications.
# Software Concepts – Summary of DOS, NOS, Middleware

<table>
<thead>
<tr>
<th>System</th>
<th>Description</th>
<th>Main Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOS</td>
<td>Tightly-coupled operating system for multi-processors and homogeneous multicomputers</td>
<td>Hide and manage hardware resources</td>
</tr>
<tr>
<td>NOS</td>
<td>Loosely-coupled operating system for heterogeneous multicomputers (LAN and WAN)</td>
<td>Offer local services to remote clients</td>
</tr>
<tr>
<td>Middleware</td>
<td>Additional layer atop of NOS implementing general-purpose services</td>
<td>Provide distribution transparency</td>
</tr>
</tbody>
</table>
Objectives of Distributed Systems

- Easily connect users to resources
  - Make remote resources available and share them in a controlled and collaborative manner - groupware, internet

- Open
  - Offers services according to standard rules that describe the syntax and semantics of those services - specified through an interface description language
  - Capable of easily interoperating with other open systems

- Transparency
  - Hide the fact that resources are distributed across a network

- Scalability
  - If it can grow in one or more dimensions without an unacceptable loss of performance
Types of Transparency

<table>
<thead>
<tr>
<th>Transparency</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>Hide differences in data representation and how a resource is accessed</td>
</tr>
<tr>
<td>Location</td>
<td>Hide where a resource is located</td>
</tr>
<tr>
<td>Migration</td>
<td>Hide that a resource may move to another location</td>
</tr>
<tr>
<td>Relocation</td>
<td>Hide that a resource may be moved to another location while in use</td>
</tr>
<tr>
<td>Replication</td>
<td>Hide that a resource is replicated</td>
</tr>
<tr>
<td>Concurrency</td>
<td>Hide that a resource may be shared by several competitive users</td>
</tr>
<tr>
<td>Failure</td>
<td>Hide the failure and recovery of a resource</td>
</tr>
</tbody>
</table>

Aiming at the highest degree of transparency may lead to a considerable loss of performance that users are not willing to accept.
Scalability Dimensions

- **Size** – add more users and resources to the systems
  - Problem – centralized, servers, data, and algorithms

- **Geographic** – users and resources may lie apart
  - Problem – synchronous communication, unreliability of WAN

- **System administration** – security and management policy to span many administrative organizations
  - Problem – protection from new domains which may have a different security policy
Scaling Technique – Reducing Communication Latency

- Avoid waiting for response to remote service requests as much as possible
  - Asynchronous communication
  - Move part of the computation that is normally done at the server to the client – shipping code is one of the options
Scaling Technique - Distribution

Taking a component, splitting it into smaller parts, and subsequently spreading those parts across the system.

An example of dividing the Internet DNS name space into zones.
Scaling Technique - Replication and Caching

- Replicates components across a distributed system
- Increases availability and load balancing
- Can hide communication latency problem if it can be replicated in different geographical location
- Caching – a special form of replication
  - Involves making a copy of a resource in the proximity of the client
  - A decision made by the client of a resource not by the owner of the resource – different from general replication
- Leads to consistency problem – degree of the problem depends on the users
Summary

- Software Concepts of Operating Systems
  - DOS, NOS, Middleware

- Objectives of Distributed Systems
  - Easily connect users to resources
  - Open
  - Transparency
  - Scalability