Distributed Systems

COMP 212

Lecture 11

Othon Michail

UNIVERSITY OF
LIVERPOOL

What is a Distributed System?

- A distributed system is:

> *A collection of independent computers that appears to its users as a single coherent system*

Goals of Distributed Systems

- Easily connect users/resources
- Transparency
- Openness
- Reliability
- Performance
- Scalability

Connecting Users and Resources

- Typical resources
  - computers, computing power, data, printers, sensors, mobile devices
- Why sharing
  - Communications, Economics/Business
  - Collaboration, Information Exchange (groupware)
- Problems (similar to societal)
  - Security
  - Coordination required
  - Fairness

Transparency in a Distributed System

- Transparent distributed system:
  - Appears to its users as if it were only a single computer system

Transparency	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may be moved to another location while in use
Replication	Hide that a resource is replicated
Concurrency	Hide that a resource may be shared by several competitive users
Failure	Hide the failure and recovery of a resource
Persistence	Hide whether a (software) resource is in memory or on disk

Degree of Transparency

Transparency is:

- Not always desirable
  - Users located in different continents (context-aware)
- Not always possible
  - Hiding failures
  - You cannot always distinguish a slow computer from a failing one
- Trade-off between a high degree of transparency and the performance of the system

Openness

Open systems:

- Offer services according to standard rules that describe the syntax and semantics of these services
- Enjoy neutral and complete specifications
  - Network protocols: standard rules for the format/contents/meaning of messages
  - Interfaces: specify services of distributed systems

Advantages:

- Interoperability
  - Different implementations can co-exist/interact by following the common standard
- Portability
  - Ability to transfer an application from one software or hardware platform to another unmodified

Open ≠Free!

Reliability

- Distributed systems should be reliable
  - Growing number of users/companies/organisations depends on them
  - Crucial applications for society and economy, highly-sensitive data
- Maximise availability
  - Fraction of time the system is usable (redundancy improves it)
- Need to maintain consistency
- Need to be secure
- Fault tolerance: need to mask failures, recover from errors
  - Many potential failures: hardware failures, software bugs, server bottlenecks, lost messages, network problems, …

Performance

- Distribution and parallelism improve efficiency
  - we can gain compared to centralised systems
- Some major challenges:
  - Communication delays
  - Lack of global coordination, global view, global synchrony, …
  - Heterogeneity
  - Trade-off with fault-tolerance/security/consistency
  - Trade-off with transparency
  - Not always cooperative: different users/vendors/stakeholders with different incentives

Scalability

Along three different dimensions:

1. Size
   - the number of users and/or processes
2. Geographical
   - maximum distance between participants
3. Administrative
   - number of administrative domains

Scalability Problems

Concept	Example
Centralised services	A single server for all users
Centralised data	A single on-line telephone book
Centralised algorithms	Doing routing based on complete information

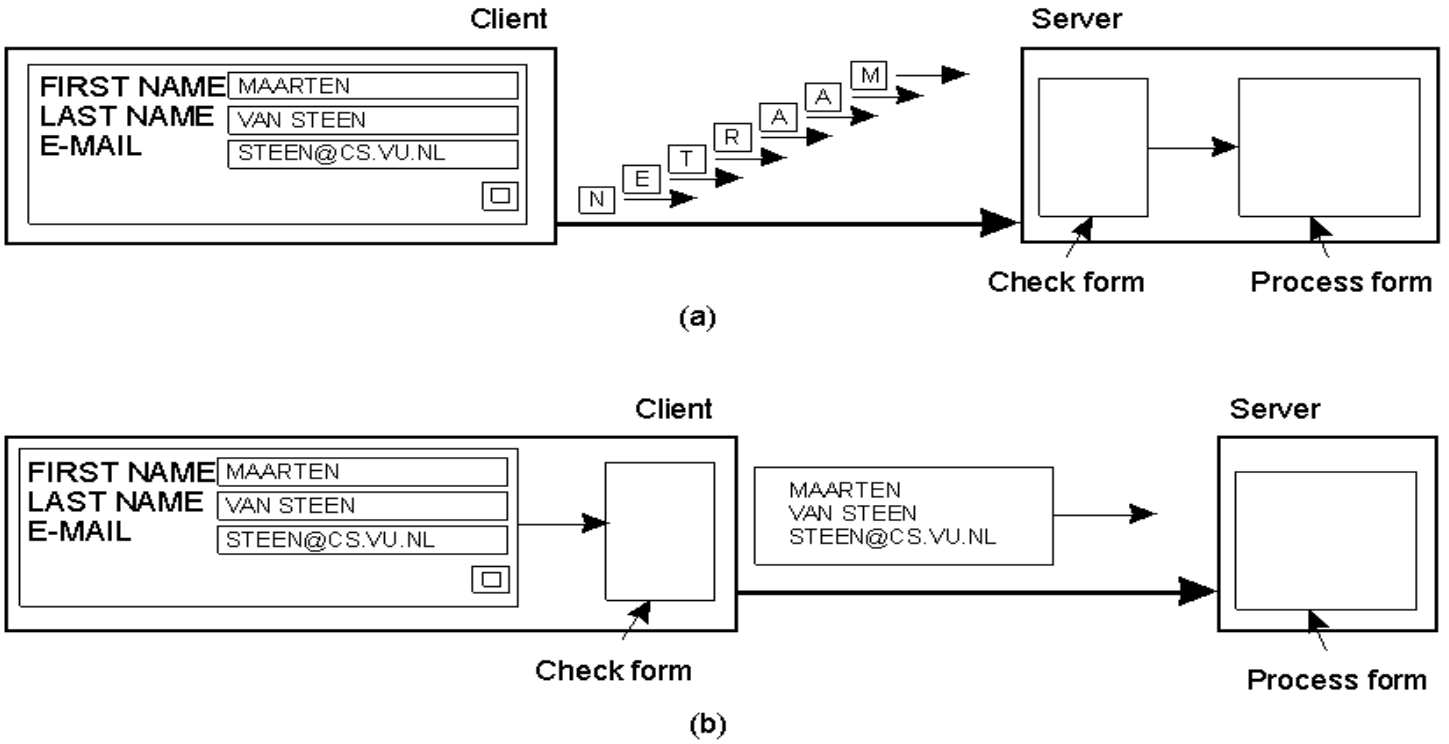
Approaches that do not scale well

Scaling Techniques

Three techniques:

1. Asynchronous communication
   - hides communication latencies
   - tries to avoid waiting for responses to remote service requests as much as possible
2. Distribution
3. Replication

Asynchronous Communication

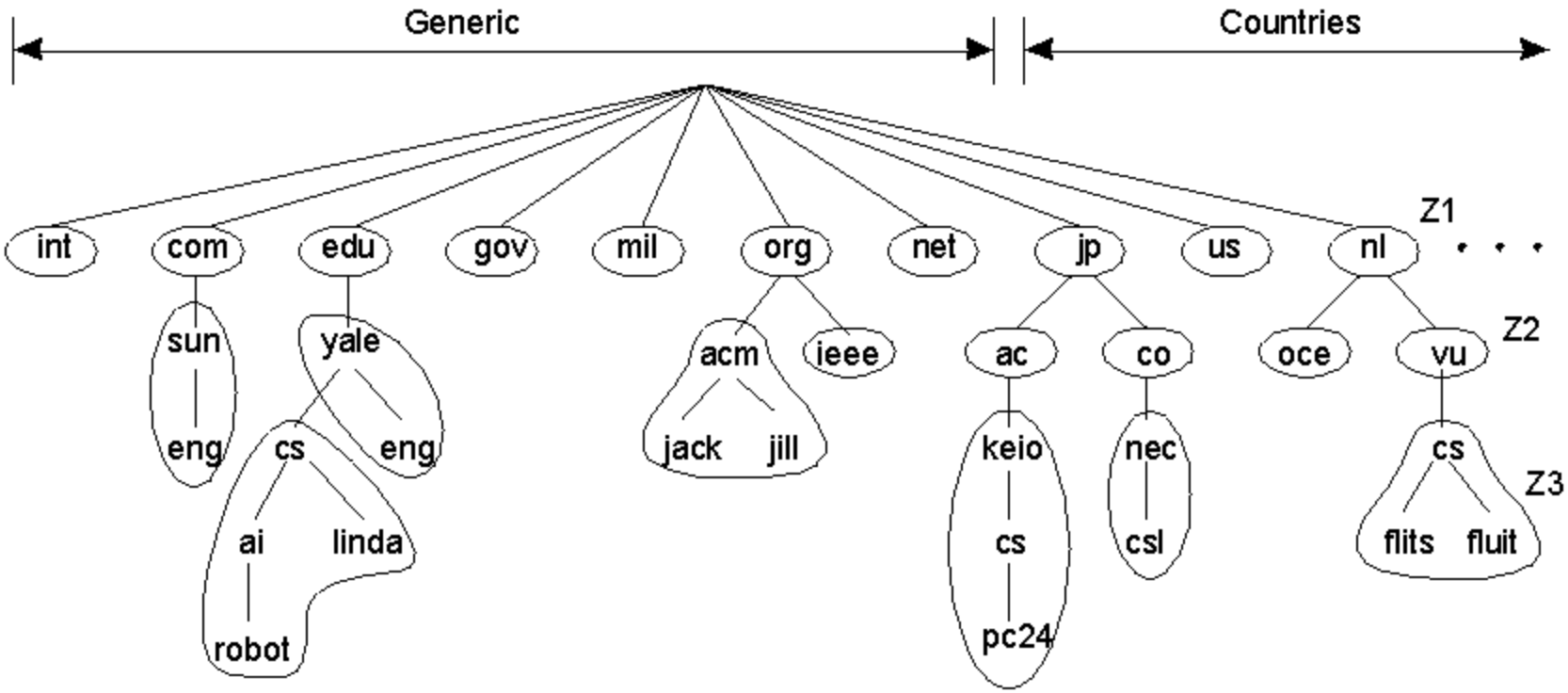


The difference between letting:

a) a server check forms or

b) a client handle them as they are being filled

Distribution



An example of dividing the DNS name space into zones.

Replication

- Increases availability
- Balances the load
- Reduces communication latency
- But causes consistency problems

Caching (client-driven)

Pitfalls

False assumptions that can make distributed systems unnecessarily complex or not functional:

1. The network is reliable
2. The network is secure
3. The topology does not change
4. Latency is zero
5. Bandwidth is infinite
6. Transport cost is zero
7. There is one administrator

Types of Distributed Systems

UNIVERSITY OF
LIVERPOOL

Types of Distributed Systems

- Many different types of distributed systems exist
- All consist of multiple CPUs
- But there are different ways in which
  1. Hardware
     - Multiprocessors, Multicomputers
  2. Software
     - Distributed Operating System, Network OS, Middleware

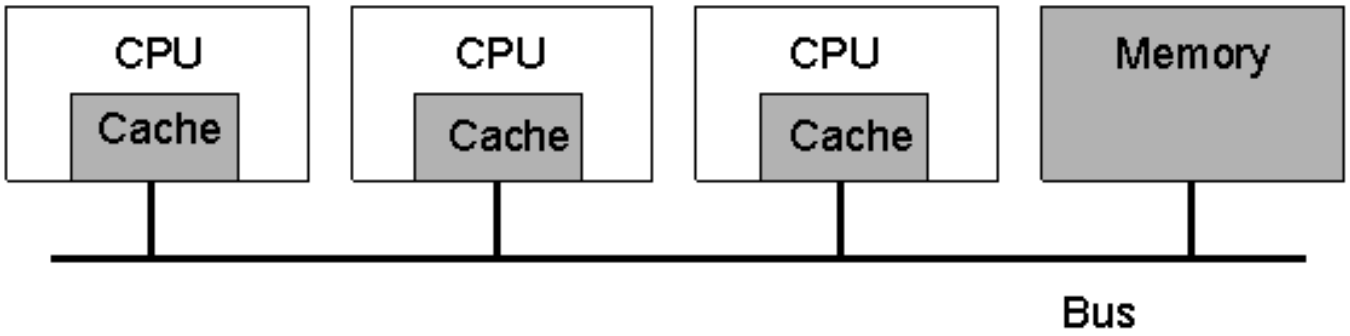
  can be organized

Hardware

Can be classified into two groups:

1. Multiprocessors
   - Shared memory
   - Several CPUs
   - All have direct access to the shared memory
2. Multicomputers
   - Each machine has its own memory
   - Network communication by message passing

Bus-based Multiprocessor System



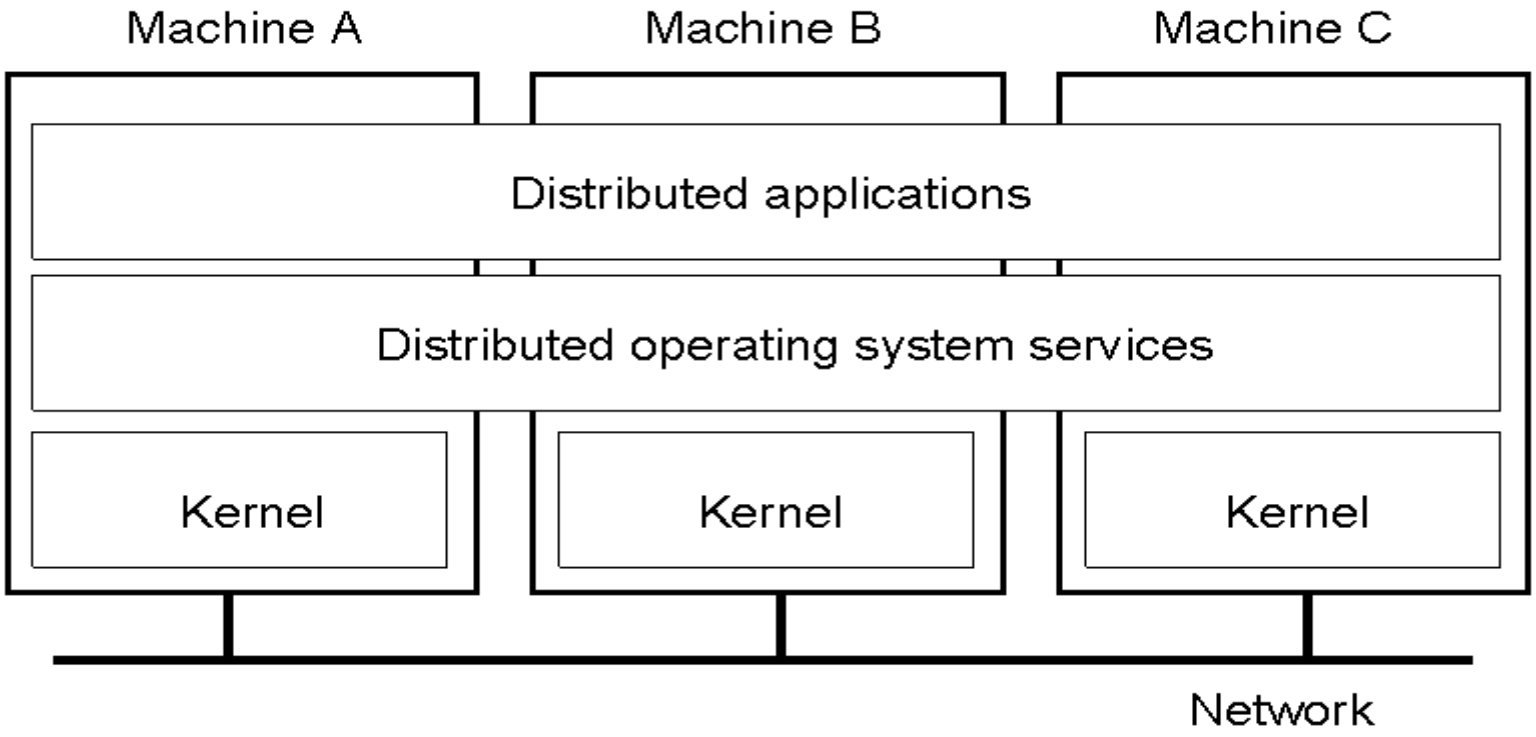
Simplified representation of a shared-memory supercomputer

- Initially for supercomputers only, but
- AMD and Intel provide 2-, 4-, and 8-processor workstations
- Alternative (e.g. for more than 256 CPUs): switching networks

Multicomputer Systems

1. Homogeneous
   - Mounted in a rack
   - Connected via a single high performance network
   - Distributed Operating Systems (DOS)
2. Heterogeneous
   - Usually of large scale and lacking global view
   - Sophisticated software is needed
   - Distributed systems provide a software layer that hides the underlying heterogeneity
   - Network Operating Systems (NOS)

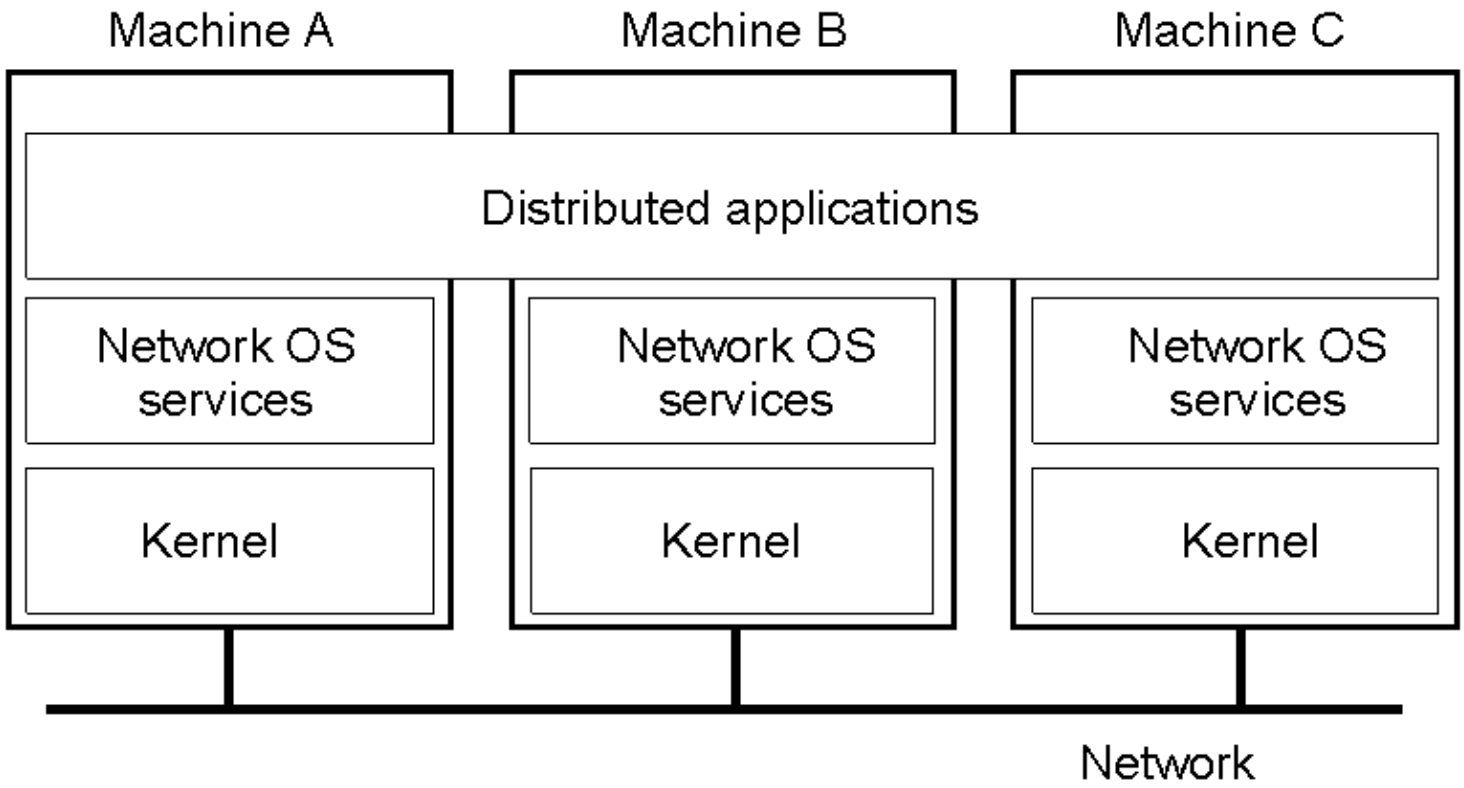
Distributed Operating Systems



General structure of a (DOS) multicomputer operating system

- Homogeneous underlying hardware
- Attempt to realise full transparency

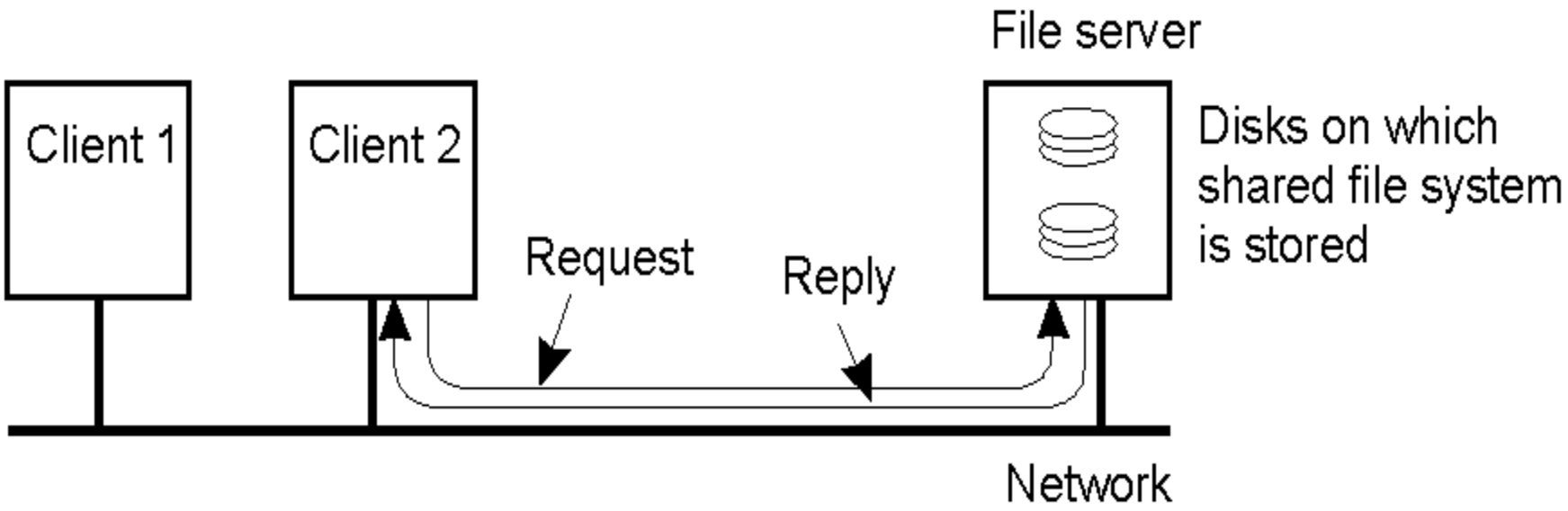
Network Operating System



General structure of a network operating system

- Heterogeneous underlying hardware
- More primitive, lack of transparency
- Easier to add/remove machines (as in the Internet)

Example

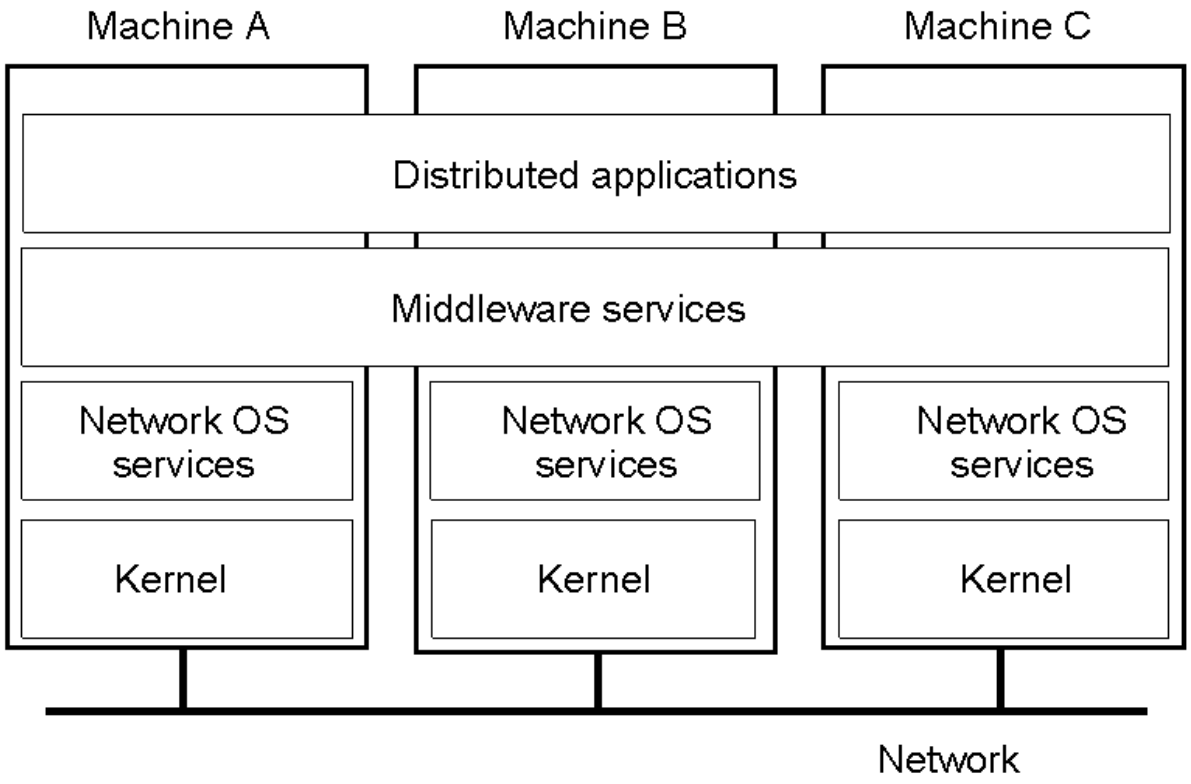


Two clients and a server in a network operating system

Middleware

- Neither NOS or DOS qualifies for distributed systems
- Is it possible to combine them to gain the best of both worlds?
  - Transparency and related ease of use of DOS
  - Scalability and openness of NOS
- Yes!
- Middleware
  - Additional layer of software on top of NOS
  - Hides the underlying heterogeneity
  - Improves transparency

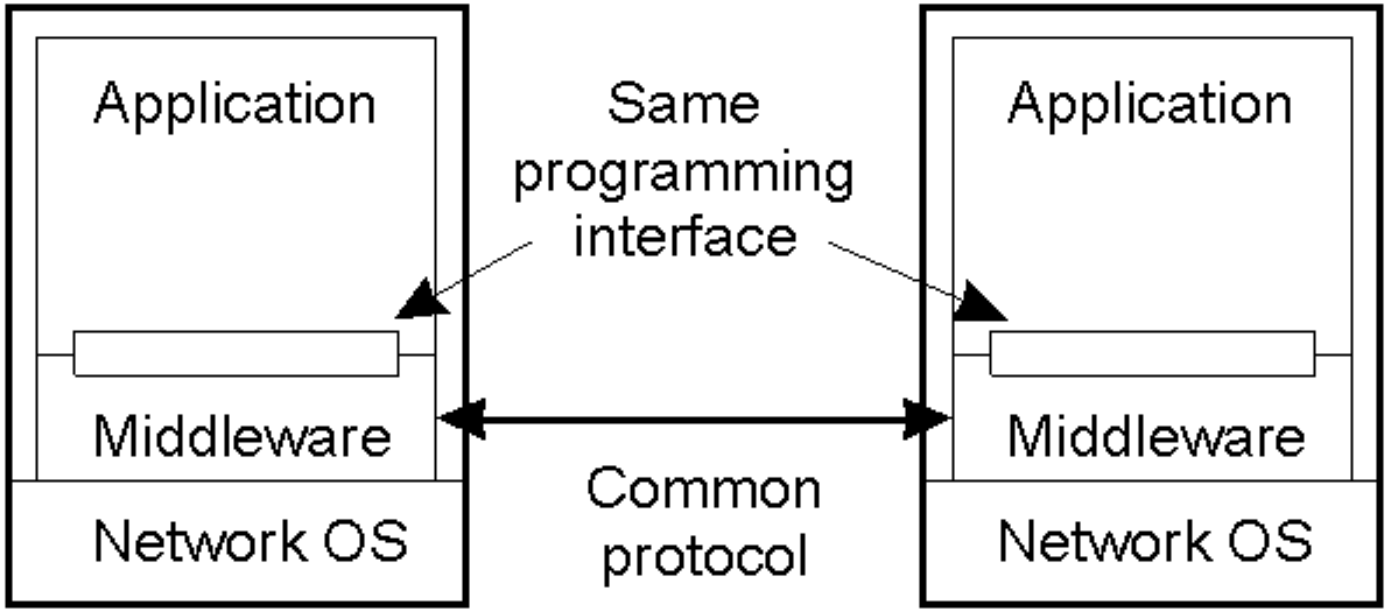
Distributed System as Middleware



A distributed system organised as middleware

- Note that the middleware layer extends over multiple machines

Middleware and Openness



- In an open middleware-based distributed system, the protocols used by each middleware layer should be the same, as well as the interfaces they offer to applications. This is a much higher level of abstraction than (for instance) the NOS Socket API.

Modelling Software Concepts: Overview

System	Description	Main Goal
DOS	Tightly-coupled operating system for multi-processors and homogeneous multicomputers	Hide and manage hardware resources
NOS	Loosely-coupled operating system for heterogeneous multicomputers (LAN and WAN)	Offer local services to remote clients
Middleware	Additional layer atop of NOS implementing general-purpose services	Provide distribution transparency

Comparing DOS/NOS/Middleware

Item	Distributed OS		Network OS	Middleware-based OS
	Multiproc.	Multicomp.		
Degree of transparency	Very High	High	Low	High
Same OS on all nodes	Yes	Yes	No	No
Number of copies of OS	1	N	N	N
Basis for communication	Shared memory	Messages	Files	Model specific
Resource management	Global, central	Global, distributed	Per node	Per node
Scalability	No	Moderately	Yes	Varies
Openness	Closed	Mostly closed	Open	Open

A comparison between multiprocessor operating systems, multicomputer operating systems, network operating systems, and middleware based distributed systems.

Summary

- Distributed Systems … autonomous computers working together to give the appearance of a single, coherent system
- They should be transparent, open, reliable, efficient, and scalable
- Unfortunately, they also tend to be complex

Summary

- Distributed systems are collections of independent computers that appear to users as a single coherent system
- They are commonly used all around the world
  - May not immediately appear as computer systems
- Main goals:
  - Easily connect users/resources
  - Exhibit transparency
  - Be open
  - Be reliable
  - Be efficient
  - Be scalable