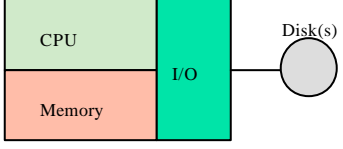



Chapter 2: Concepts and Architectures



Traditional Computer Architecture

Chapter 2 Concepts and Architectures

1




Computer System Architectures

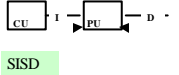
- Flynn, 1966+1972 classification of computer systems in terms of instruction and data stream organizations
- Based on Von-Neumann model (separate processor and memory units)
- 4 machine organizations
 - SISD - Single Instruction, Single Data
 - SIMD - Single Instruction, Multiple Data
 - MISD - Multiple Instruction, Single Data
 - MIMD - Multiple Instruction, Multiple Data

Chapter 2 Concepts and Architectures

2

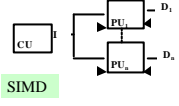


Flynn Architectures (1)



Serial Processor

SISD



Array Processor

SIMD

CU – control unit


PU – processor unit

I – instruction stream

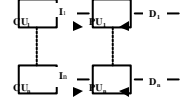
D – data stream

Chapter 2 Concepts and Architectures

3

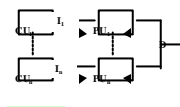


Flynn Architectures (2)



Multiprocessor and Multicomputer

MIMD




MISD

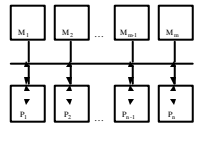
No real examples - possibly some pipeline architectures

Chapter 2 Concepts and Architectures

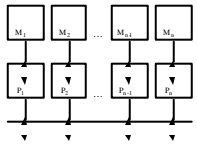
4



Common Bus based




(1) Shared memory



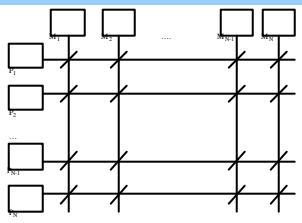
(2) Local memory

Chapter 2 Concepts and Architectures

5



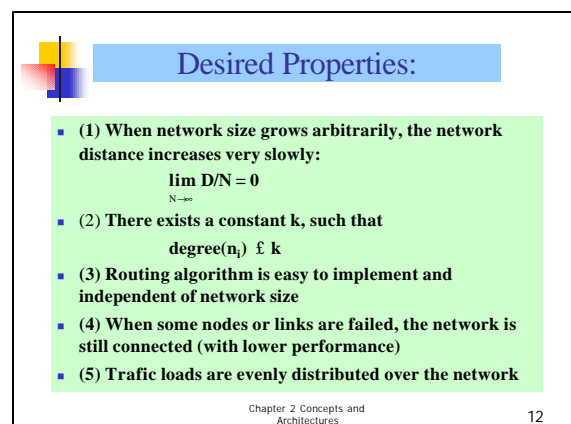
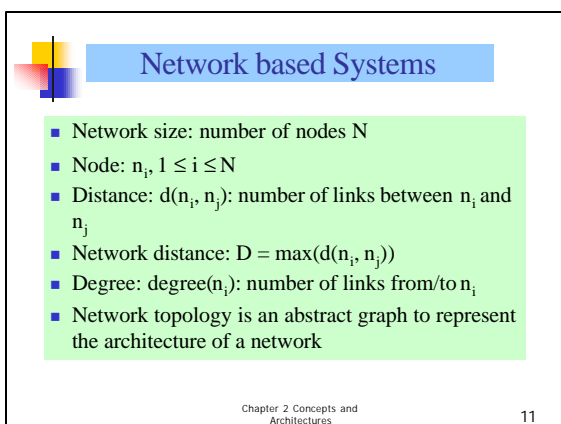
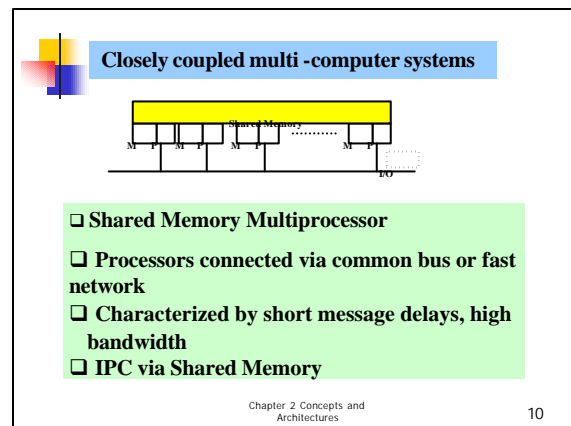
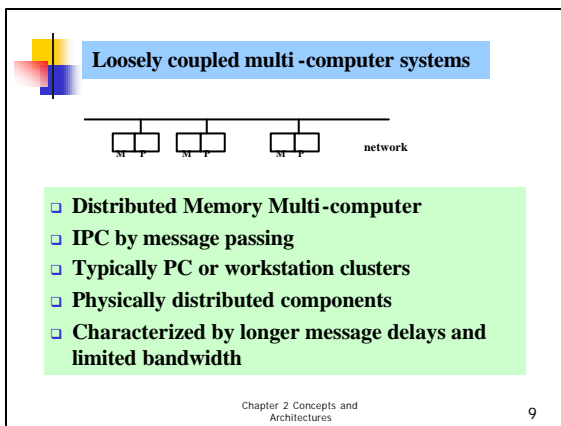
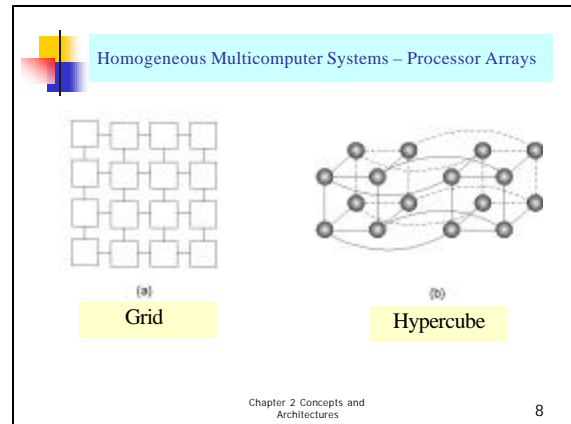
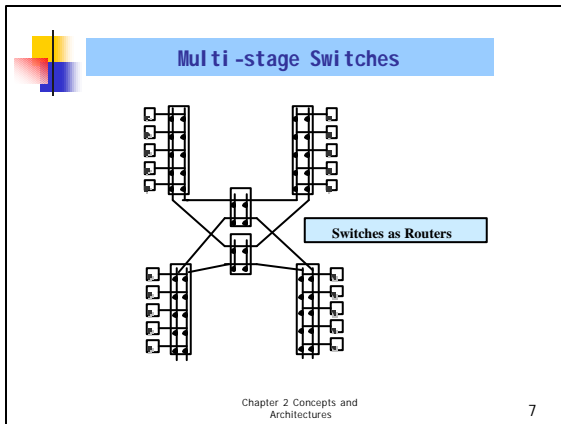
Crossbar switch



through N^2 switches to access memory banks

Chapter 2 Concepts and Architectures

6



Typical network topologies:

The diagram illustrates six common network topologies:

- star**: A central node connected to multiple peripheral nodes.
- ring**: Nodes connected in a closed loop.
- B-tree**: A hierarchical tree structure with multiple levels.
- complete**: Every node is connected to every other node.
- regular**: A grid-like structure with nodes connected to a fixed number of neighbors.
- arbitrary**: A random or irregular connection pattern.

Chapter 2 Concepts and Architectures 13

Evaluation of network topologies:

	star	ring	B-tree	complete	regular	arbitrary
$\lim_{N \rightarrow \infty} D/N = 0$	yes $D = 2$	no $D = N-1$	yes $D = 2 \log N$	yes $D = 1$	yes $D = \sqrt{N}$	no not know
$C = d(n_1)$	no	yes $C = 2$	yes $C = 3$	no	yes $C = 4$	no not know
routing	easy	easy	easy	easy	easy	hard
connectivity	bad	bad	not good	best	good	no not know
even traffic	no	yes	no	yes	yes	no not know

Chapter 2 Concepts and Architectures 14

Software Concepts

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

- DOS (Distributed Operating Systems)
- NOS (Network Operating Systems)
- Middleware

Chapter 2 Concepts and Architectures 15

Uniprocessor Operating System

The diagram shows a uniprocessor operating system architecture. At the top is the **Application** layer, which contains **process**, **device**, **file**, and **memory** components. Below this is the **OS** layer, which manages these resources. At the bottom is the **Computer** hardware layer. A green box at the bottom states: **virtual machine: multiple concurrent processes running under a uniprocessor OS.**

Chapter 2 Concepts and Architectures 16

Distributed Operating System

The diagram illustrates a distributed operating system architecture. It shows **Distributed Applications** at the top, which are divided into **process**, **device**, **file**, and **memory** components. Below these is the **OS** layer, which manages the resources. At the bottom is the **hardware...** layer, which is connected to a **net** (network). A green box at the bottom states: **Tightly-coupled operating system for multi-processors and homogeneous multi-computers. Strong transparency.**

Chapter 2 Concepts and Architectures 17

DOS: characteristics (1)

Distributed Operating Systems

- Allows a multiprocessor or multicomputer network resources to be integrated as a single system image
- Hide and manage hardware and software resources
- provides transparency support
- provide heterogeneity support
- control network in most effective way
- consists of low level commands + local operating systems + distributed features
- Inter-process communication (IPC)

Chapter 2 Concepts and Architectures 18

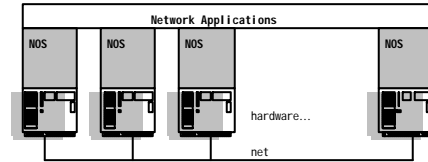
DOS: characteristics (2)

- remote file and device access
- global addressing and naming
- trading and naming services
- synchronization and deadlock avoidance
- resource allocation and protection
- global resource sharing
- deadlock avoidance
- communication security
- no examples in general use but many research systems: Amoeba, Chorus etc. see Google "distributed systems research"

Chapter 2 Concepts and Architectures

19

Network Operating System



Loosely-coupled operating system for heterogeneous multi-computers (LAN and WAN). Weak transparency.

Chapter 2 Concepts and Architectures

20

NOS: characteristics

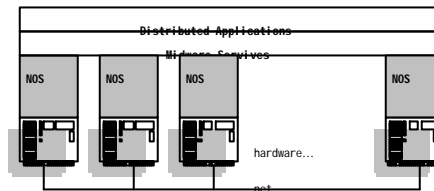
Network Operating System

- extension of centralized operating systems
- offer local services to remote clients
- each processor has own operating system
- user owns a machine, but can access others (e.g. rlogin, telnet)
- no global naming of resources
- system has little fault tolerance
- e.g. UNIX, Windows NT, 2000

Chapter 2 Concepts and Architectures

21

Middleware System



Additional layer on the top of NOS implementing general-purpose services. Better transparency.

Chapter 2 Concepts and Architectures

22

Middleware Examples

Examples: Sun RPC, CORBA, DCOM, Java RMI (distributed object technology)

Built on top of transport layer in the ISO/OSI 7 layer reference model: application (protocol), presentation (semantic), session (dialogue), transport (e.g. TCP or UDP), network (IP, ATM etc), data link (frames, checksum), physical (bits and bytes)

Most are implemented over the internet protocols

Masks heterogeneity of underlying networks, hardware, operating system and programming languages – so provides a uniform programming model with standard services

3 types of middleware:

- transaction oriented (for distributed database applications)
- message oriented (for reliable asynchronous communication)
- remote procedure calls (RPC) – the original OO middleware

Chapter 2 Concepts and Architectures

23

Comparison between Systems

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	Closed	Open	Open

Chapter 2 Concepts and Architectures

24

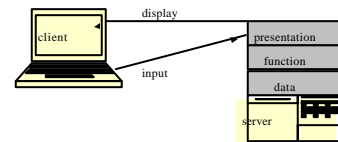
System Platforms

- **client/server**: consider users as clients, and services provider as servers.
- **browser/server**: general purpose client interface, easy to use.

Chapter 2 Concepts and Architectures

25

Terminal/Machine Model

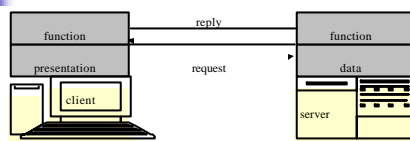


- (1) presentation logic: how to interact?
- (2) function logic: how to implement function?
- (3) data logic: how to maintain, update and protect data?

Chapter 2 Concepts and Architectures

26

Client/Server Model

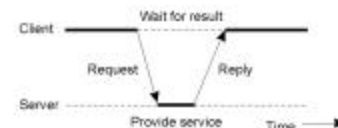


presentation: GUI: Graphic User Interface.
function: implemented in server side or partly in client side for efficiency.
data: DBMS.

Chapter 2 Concepts and Architectures

27

Timing interaction between client and server

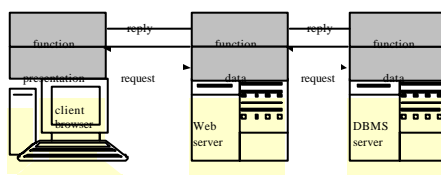


Client/Server interaction can be implemented by either message passing or RPC(remote Procedure Call).

Chapter 2 Concepts and Architectures

28

Browser/Server Model



Implemented as a three-tiered architecture

Chapter 2 Concepts and Architectures

29

Browser/Server Model (2)

- easy to use
- simply system development, update
- easy to be standard
- weak interactive
- weak security
- lower efficiency (compare with client/server)

Chapter 2 Concepts and Architectures

30

