Introduction to Operating System

Prof P.C.P. Bhatt

OS Services Support Applications on Computers

We often use computers for a variety of applications which require some logistical system support. A few typical applications are listed below:

- Document design
- Accounting
- E-mail
- Image Processing
- Games

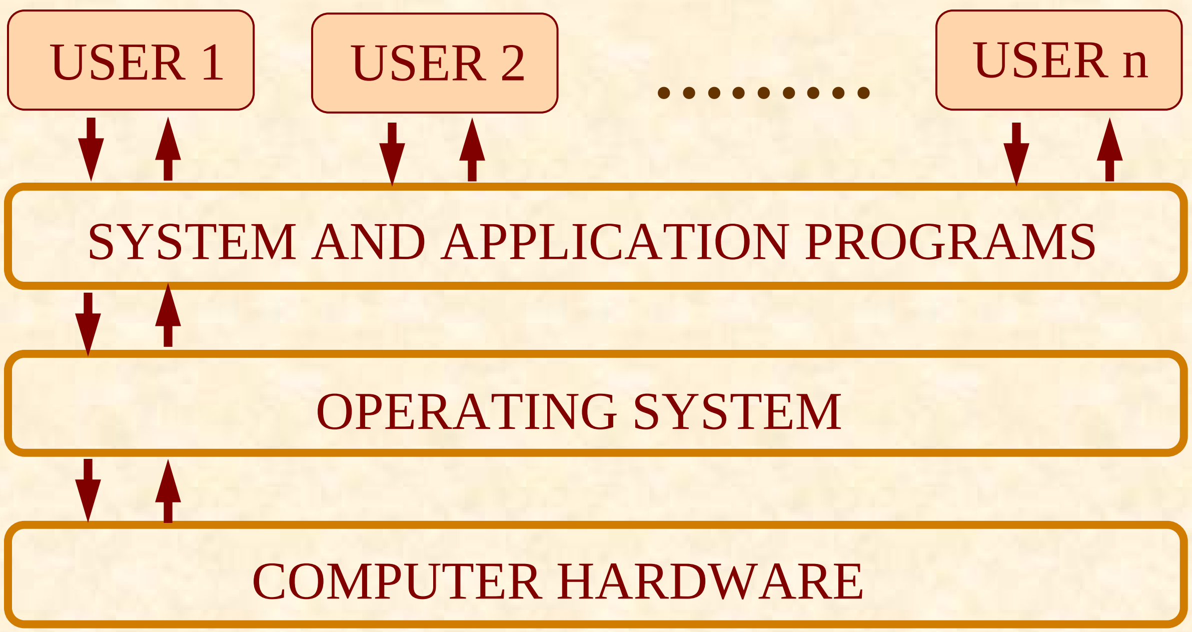
OS support is *application neutral* and *service- specific.*

Operating System : Definition

- It is the *software layer*, nearest to hardware which facilitates *launching* of all the other *software utilities* and applications.
- OS provides wide range of *generic data services.*
- *Manages* keyboard, display, processor, memory and other devices.
- *Schedules* input, output and data processing.

User and OS

- OS facilitates use of resources by *hiding local details* and *presenting an interface* which is convenient to use.
- For instance : computer games, e-mail, browsing or preparing documents are applications launched by simply *clicking on cue icons.* How easy it is !



User and System View of OS

- *User perspective* : ease of usage is the main consideration.
- *System perspective* : efficiency in usage of resources is the main consideration.

As a provider of resources - OS must have a *policy and a control program* to regulate the allocation of resources.

Systems in Early 60s

- Main frames: Housed in *Computer center.*
- Users submitted *"jobs"* as deck of punched cards.
- Job batches of *"Fortran Jobs, Cobol jobs".*
- Jobs executed sequentially – *one job at a time.*
- *"Job header"* control cards were used to define users' need of resources.

Late 60s: Multiprogramming

- Sequential processing *wasteful.*
- Processor and Memory *not fully utilized*.
- Processor *idles* during Input output.
- Solution: Allow multiple programs to *reside* in the main memory.
- When one program *engages* I/O the other can use the processor.

Still in Late 60s : Time Sharing

- Multiple *users access* the system.
- Each one gets a *time-slice* for his job.
- Users get an illusion as if he has the *whole system for himself.*
- Time shared systems must *inherently support multiprogramming.*

OS : Design Considerations

- *To achieve higher throughput* : I/O could be overlapped with processing.
- In 70s we witnessed emergence of high speed disks as *"secondary storage".*
- Address space enhanced to achieve access to *"virtual address space".*
- The Strategy: *"swap out"* what is not needed immediately, *"swap in"* what is required for current processing.

Systems in Mid to Late 70s

- Systems *supported multiple terminals*.
- The metaphor: instead of *"user going to compute"* the *"computing should be brought"* to the user.
- That metaphor led to *remote terminal operations.* It also led to *micro-computer revolution* which set the scene for *launch of PCs*.

OS in 70s and 80s

- Major contribution by *Bell Labs : Unix.*
- Unix (1972) supported *time-shared multi-user operation.*
- With Micro-computers on the scene *small foot-print OS like CP-M (1980).*

Client-Server Paradigm (80s)

- Project *“Athena” at MIT* developed the *X-Clients.*
- Also, a *“window”* as a virtual terminal gave a user a capability to *launch multiple applications* from the same terminal.
- A window *“client”* seeks a service from a *“server”.*
- A *“compute server”* could be sought for processing. A *“file server”* could be accessed for *“file access”.*

Early to Mid 80s : PC Arrives

- Need felt to distribute IO processing : led to the *development of BIOS.*
- Also, led to graphic drivers like *EGA, VGA cards.*
- *Networking support* developed.
- *Unix (a command oriented OS)* also developed Networking support.
- MAC developed *"drag and drop"* and icon based *"launch"* for applications.

Parallel Processing – Mid 80s

- Applications like weather prediction, medical image processing require computing power *in excess of what a single processor can provide.*
- Leads us to using *"multi-processor"* architectures.
- *"Tightly coupled"* and *"loosely coupled"* multi-processor systems.
- One advantage one gets is *fail-safe operation* in addition to the *massive computing capability.*
- *Symmetric / asymmetric multi-processing* refers to *uniform OS or heterogeneous OS on interconnected systems*

So what does an OS do?

- Power On Self Test *(POST)*.
- Resource management.
- Support for *multi-user.*
- Error Handling.
- *Communication support* over Network.
- (Optional) *Deadline support* so that safety critical application run and *fail gracefully.*

Operating System Facilities

- User *access* to the system
- *Storage* and *management* of information
- *Protection* of information against accidental and intentional misuse
- *Support* for data processing activities
- *Communication* with I/O devices
- *Management* of all activities in a transparent manner.

Operative Environment for RTOS

The Key Factors

- Sensor
- Control Settings
- System
- Environment
- Interface

Typical Operating Sequence

1. Sense an event
2. Process the data
3. Decide on an action
4. Take corrective action

An Example of Real Time Control Application

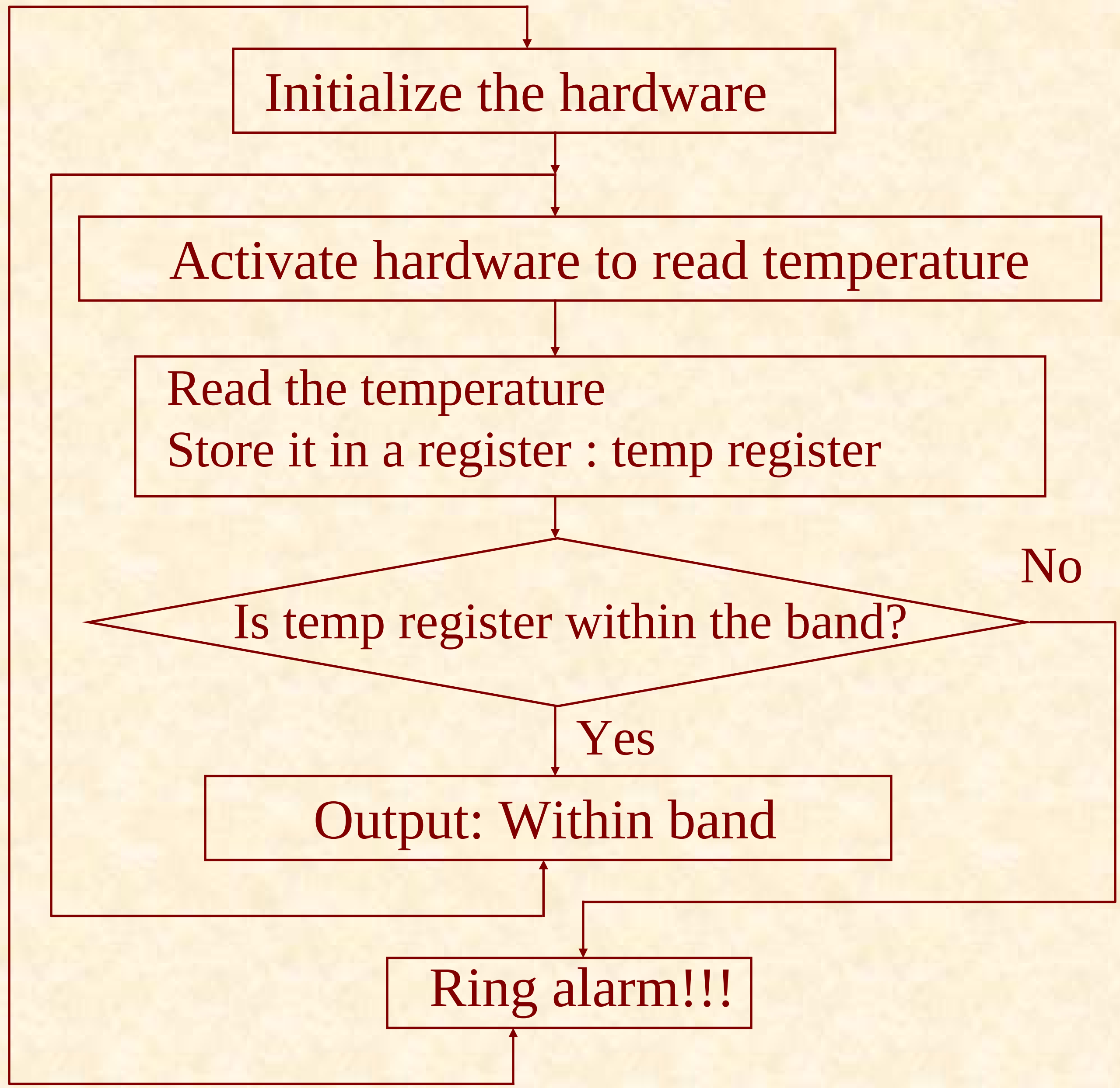
- *Scenario* : A temperature monitoring chemical process.
- *What we need* : A supervisory program to raise an alarm when temperature goes beyond a certain band.
- *The desired sequence of operational events* : Measure input temperature, process the most recent measurement, perform an output task.

Monitoring

Input

Decision

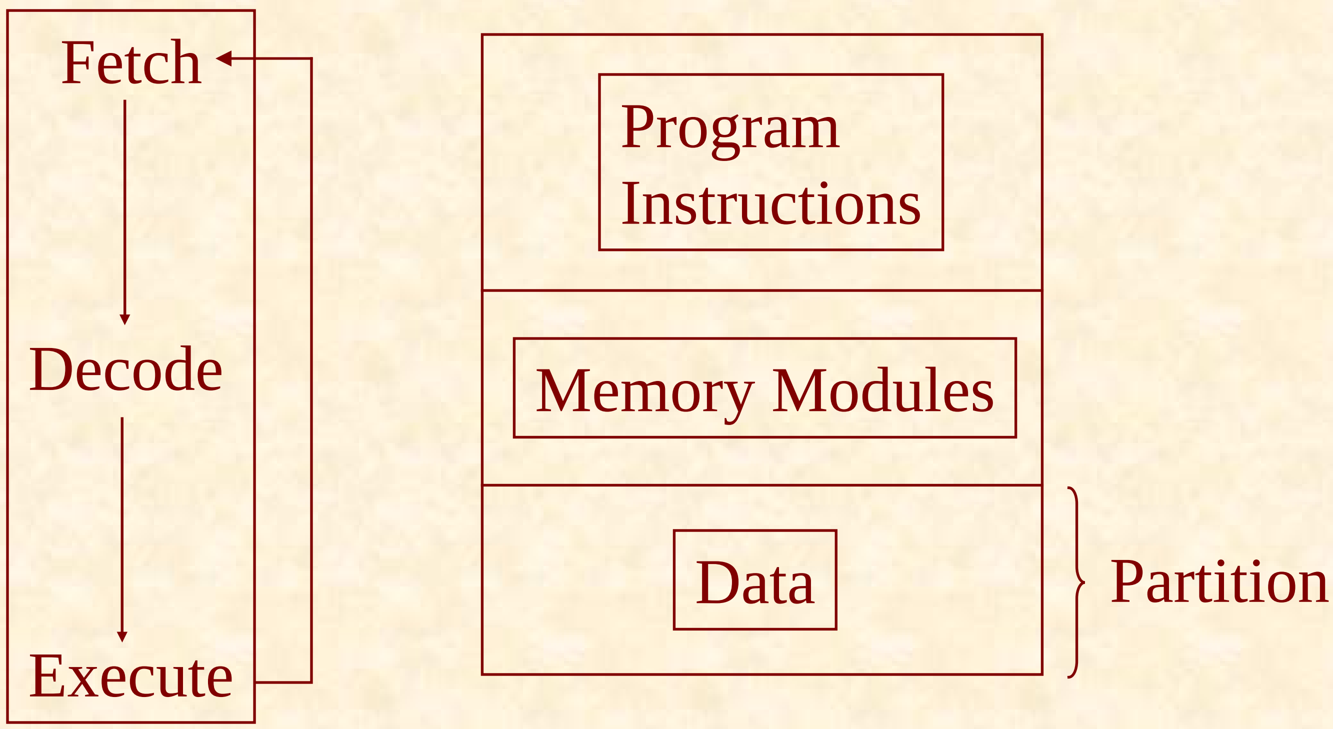
Output



The Example Continues….

- *Monitoring* – this phase initializes and activates the hardware.
- *Input* – reads the values from sensors and stores it in register.
- *Decision* – checks whether the readings are within the range.
- *Output* – responds to the situation.

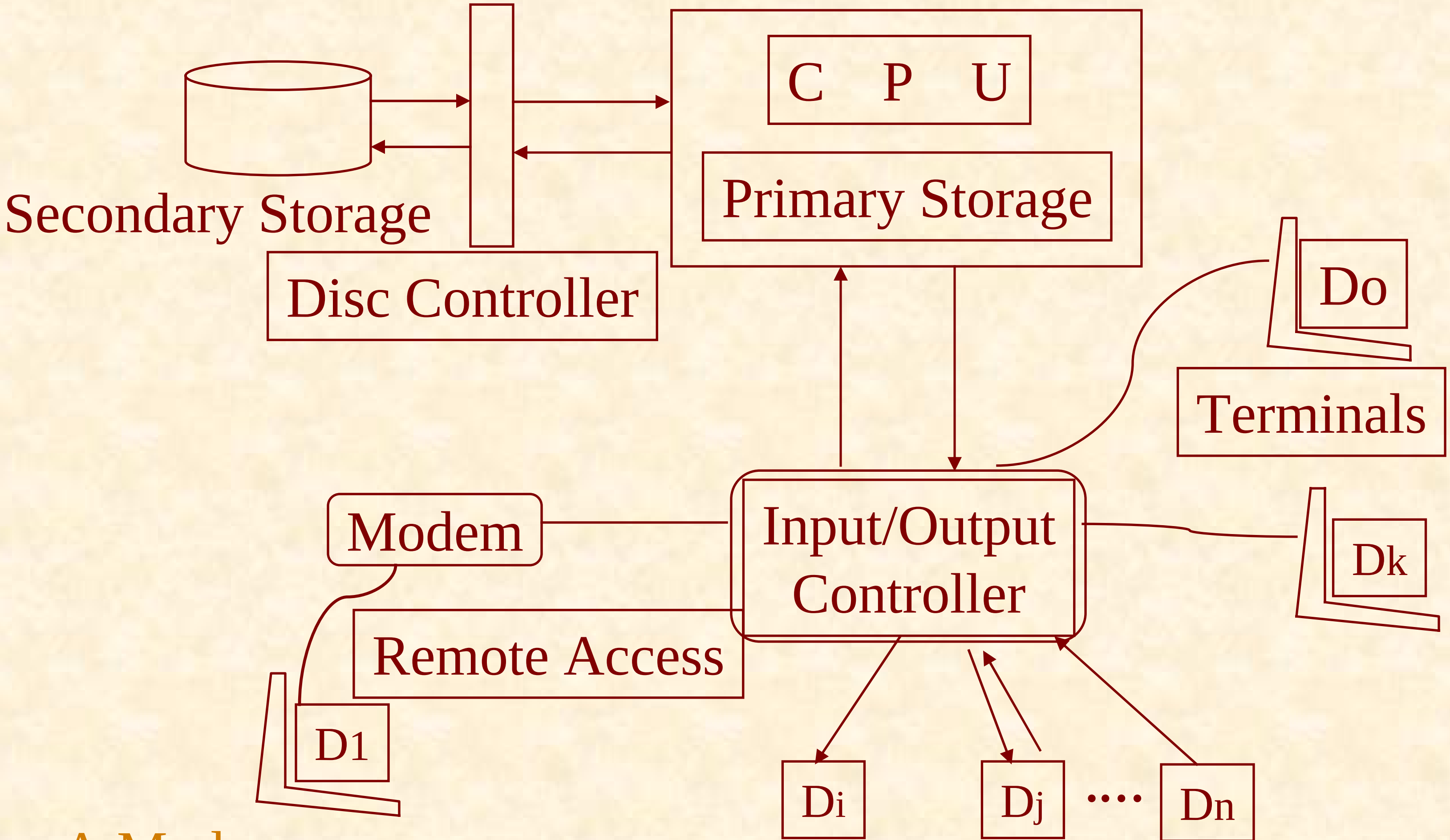
The Von Neumann Operational View



Operational Overview

- Processor – *executes programs, schedules* and *allocate* processor time.
- Memory – *stores programs*, and *supports mechanisms to access* data
- Input output devices – supports all *input* and *outpu*t operations
- Communication mechanisms - with *devices external* to the system
- Mutual exclusion – schedule the usage of *shared device* and *fair access*
- Shell of an OS
- Human computer interface (*HCI/CHI*)

A Physical Overview of a Computer System

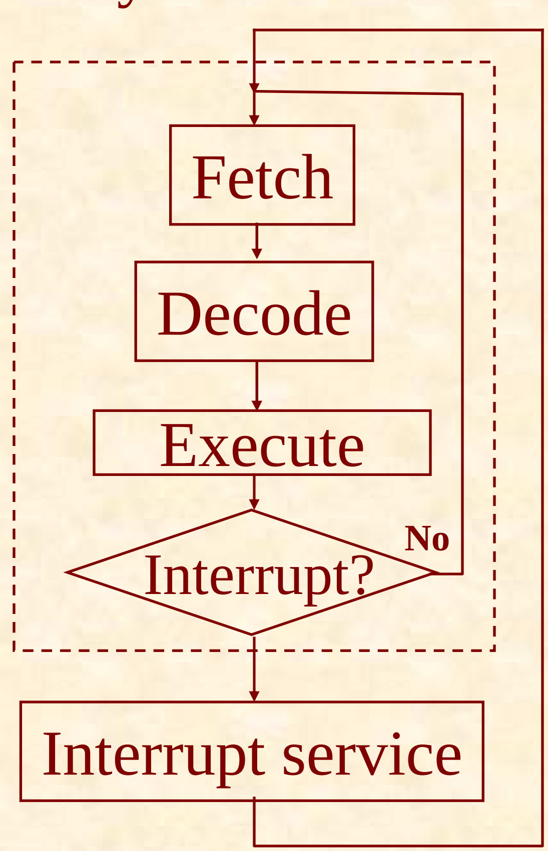


A Modern Computer System

D1 to Dn are I/O devices

Input Output Devices

The normal instruction cycle



Servicing an Interrupt

Processes and Tools

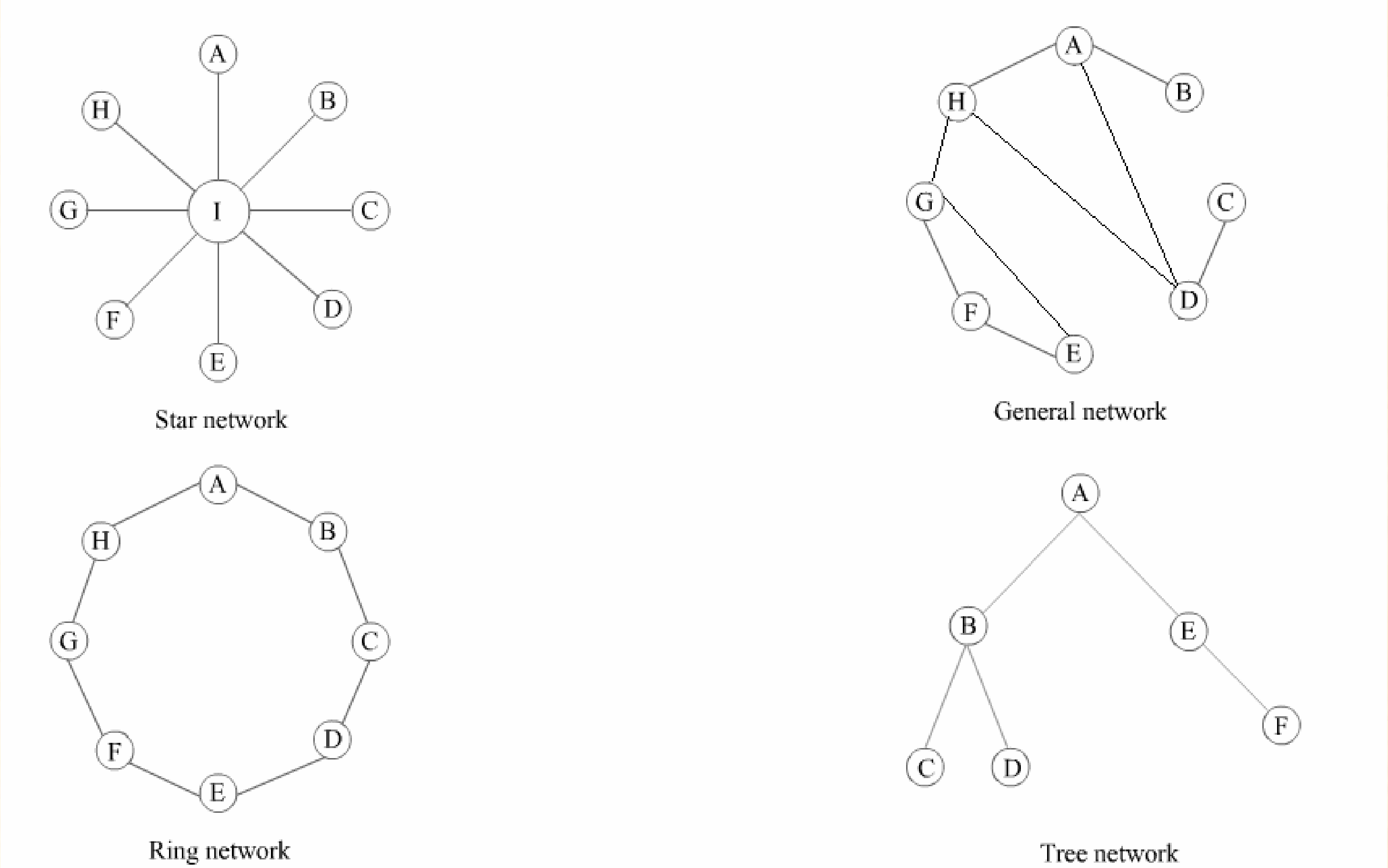
- Program in execution is called a *process.*
- Interprocess communication forms the basis of *distributed computing.*
- The machine seeking the service is *client* and the machine offering the service is *server.*
- OS provide many general purpose utilities as a set of packaged *tools.*

Impact of Networking

- Computers communication NW required OS to be NW aware *enabling remote resource sharing.*
- Programs, files or data could be *moved around* and *accessed.*
- Lead to protocols like *FTP, telnet, RPC.*
- Internet required *HTTP and browser support.*

Network Topologies

- Some NW topologies



Other Trends in Computing

- *Web based computing.* Essentially web based services made available.
- Systems like *Web sphere, web logic* provide such services.
- Makes the prophetic statement by *Scott McNealy,* CEO of SUN, *"Network is the computer"* appear true.

Some Notable Contributions

- University of Manchester: One level store and Manchester encoding.
- Dartmouth College: Time sharing systems
- MIT: Multics and Proj. Athena, Client server arch.
- Bell Laboratories and UC Berkeley : Unix and Networking Software.
- Apple and Microsoft: Icon based HCI