

# **CSCE 351**

## **Operating System Kernels**

---

### **History of Operating Systems & Basic Operating System Concepts**

**Prof. Steve Goddard**  
***goddard@cse.unl.edu***

*<http://www.cse.unl.edu/~goddard/Courses/CSCE351>*

1

## **What is an Operating System?**

---

- ◆ A program
- ◆ An interface
- ◆ A programming environment
- ◆ A resource manager
- ◆ A service provider

2

## Why do we need operating systems?

---

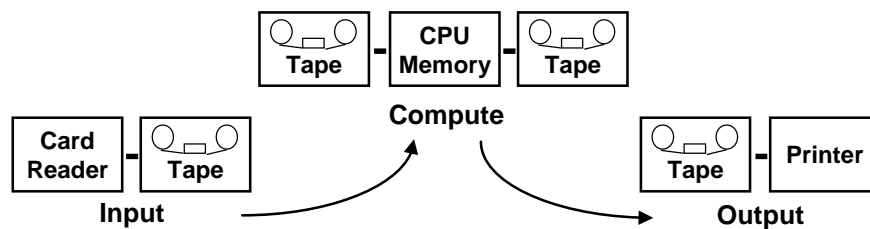
- ◆ They provide a high-level abstraction of physical resources
- ◆ They allow sharing of limited or expensive physical resources

3

## A brief history of Operating Systems

---

- ◆ Hand programmed machines ('45-'55)
- ◆ Batch processing/Off-line processing ('55-'65)



4

## A brief history of Operating Systems

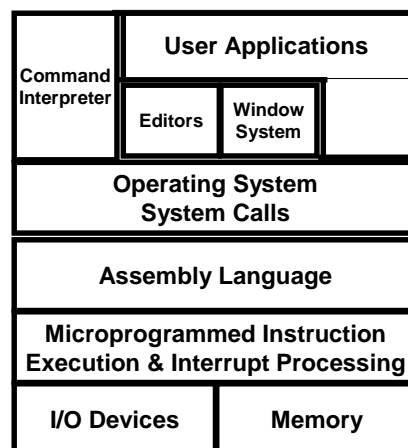
---

- ◆ On-line processing ('62-'69)
- ◆ Multiprogramming ('65-'80)
- ◆ Timesharing ('70- )
- ◆ Personal computing ('80- )

5

## What is an Operating System Today?

---



6

## Basic Operating System Functions

---

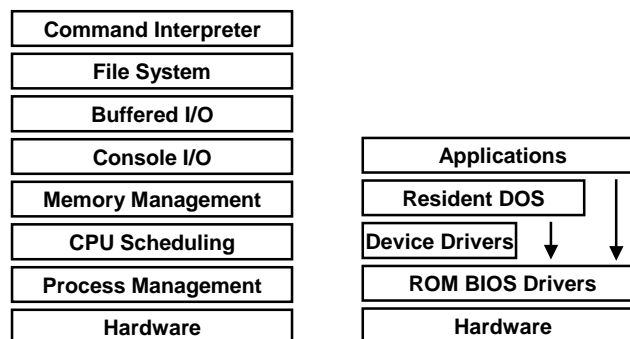
- ◆ Device management
  - » Programmed I/O
  - » DMA
- ◆ Secondary storage management
- ◆ File systems
- ◆ Memory management
- ◆ Process management
- ◆ Protection & Security
- ◆ Network communications

7

## Operating System Structures

---

- ◆ Layered



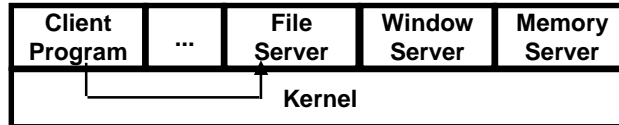
- ◆ Monolithic

8

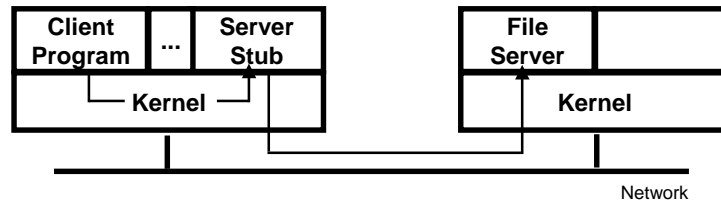
## Operating System Structures

### ◆ Client/Server or “Microkernel”

» Centralized

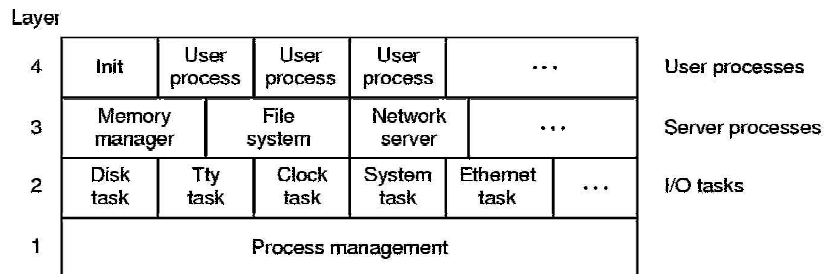


» Distributed



9

## MINIX Structure



◆ Is it layered, monolithic, a microkernel or what?

10

## Basic Operating System Concepts

---

- ◆ A process
  
- ◆ An address space or context

11

## Basic Operating System Concepts: A system call

---

### Example: Programmed I/O

```
process UserProg
begin
:
  read(file, buffer,#bytes)
:
end UserProg

procedure Read(file, buff, bytes)
begin
:
  Read(file, buff,bytes)
:
end Read

_Read:
LOAD r1,@SP+2
LOAD r2,@SP+4
LOAD r3,@SP+6
TRAP Read_Call
```

The diagram illustrates a system call. An arrow points from the `read(file, buffer,#bytes)` line in the `UserProg` process to the `Read(file, buff, bytes)` line in the `Read` procedure. A curved arrow points from the `end Read` line back to the `_Read:` label, indicating the return path.

12