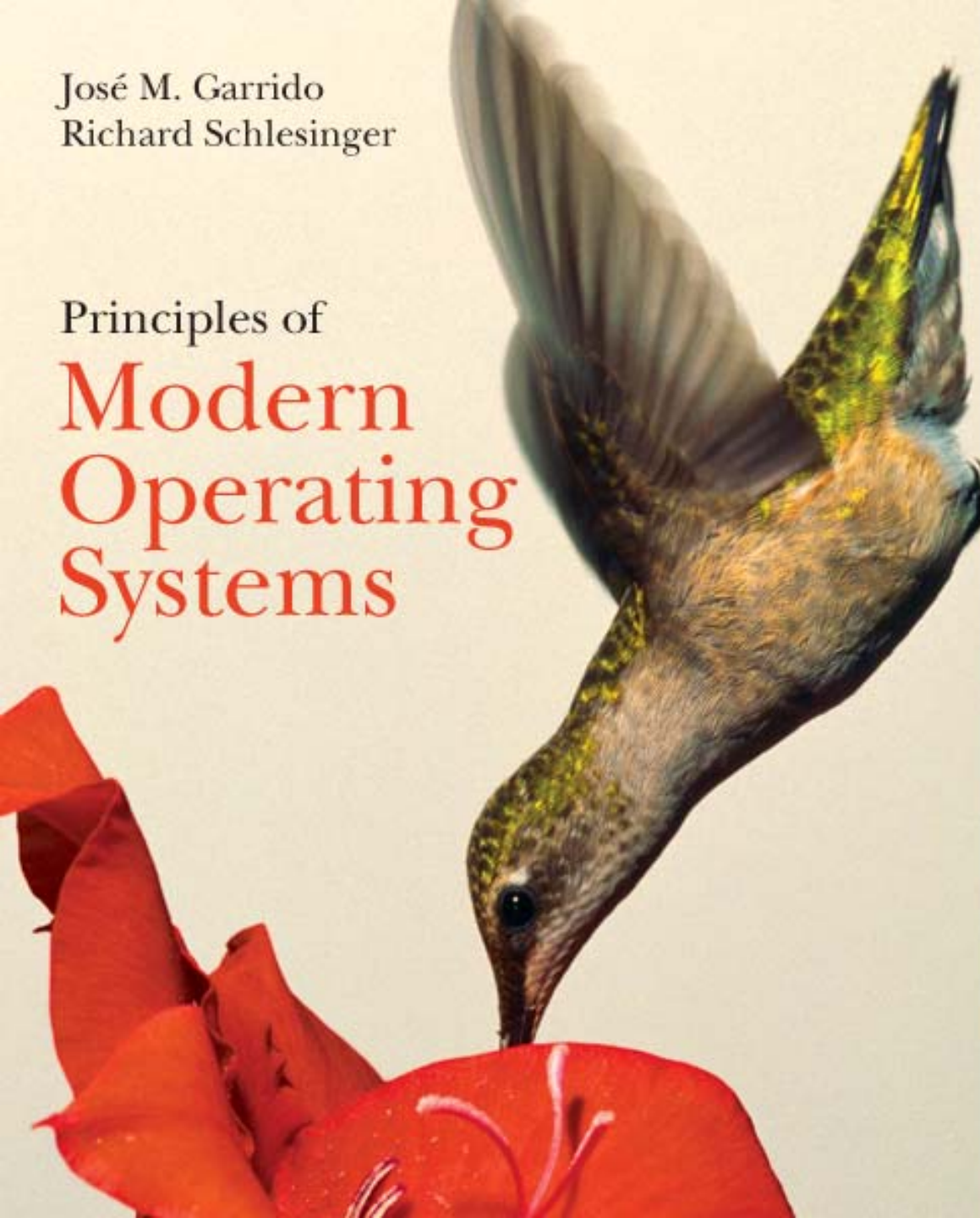


José M. Garrido
Richard Schlesinger

Principles of
**Modern
Operating
Systems**



Chapter 4

Systems with Multi- programming

Multiprogramming - Review

- An operating system can support several processes in memory.
- While one process receives service from the CPU, another process receives service from an I/O device and the other processes are waiting in some queues.
- The number of processes that the system supports is called the degree of multiprogramming.

Multi-Programming

- In most computer systems, the I/O controllers enable a system to overlap device I/O operation with processor (CPU) operation.
- This results in a more efficient utilization of the system facilities
- Several processes can be in memory at a time, one receiving CPU service and another receiving I/O service.

Requirements for Multiprogramming

- The OS must allocate the CPU and other resources to the various processes in such a way that the CPU and other active resources are maintained busy the longest period possible.
- If there is only one CPU in the system, then only one process can be in execution at any given time.
- The other processes are ready, waiting for CPU service. Processes also request access to passive resources, such as memory locations.

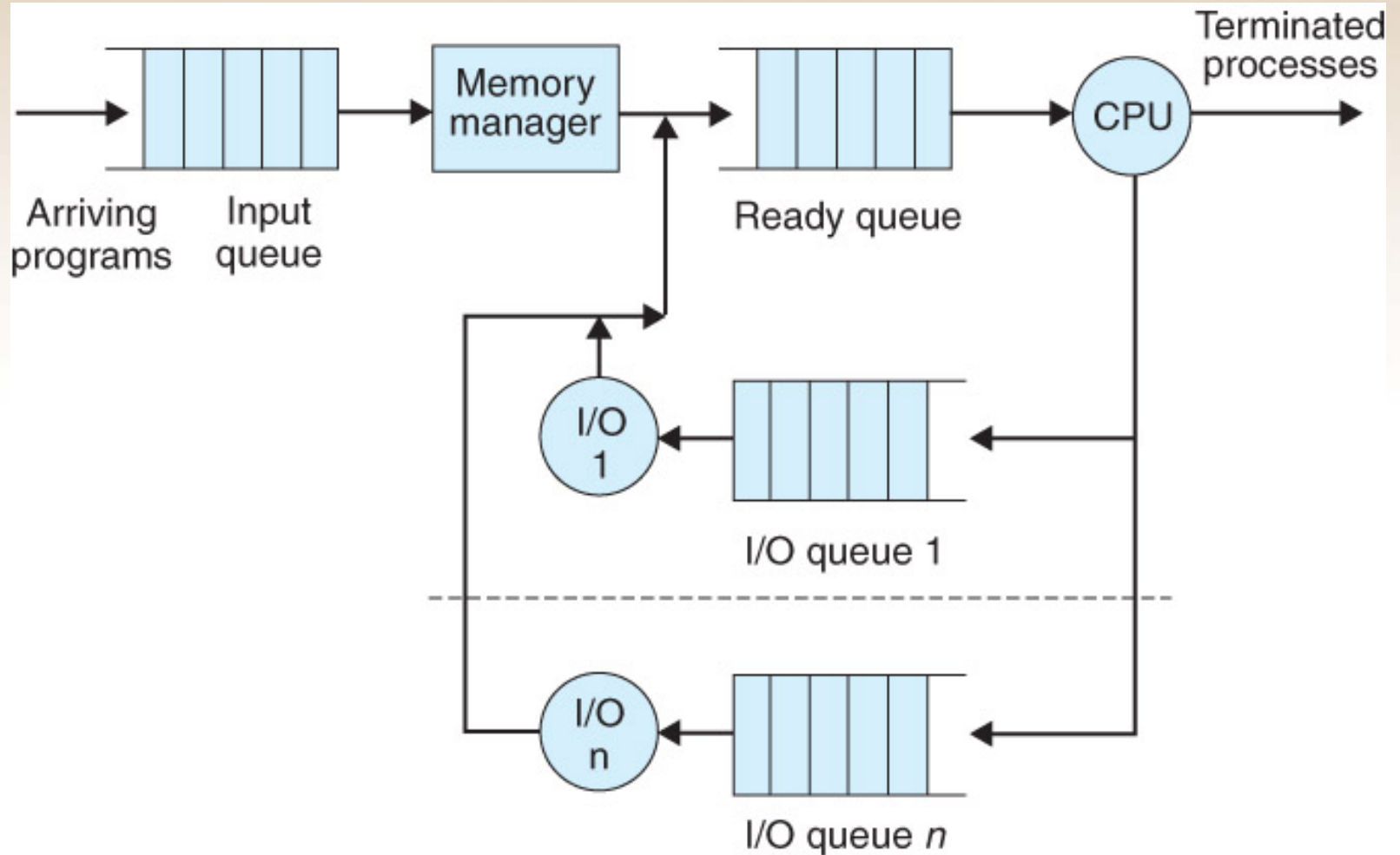
Process Service Requests

- A process requests CPU and I/O services at various times and usually will have to wait for these services.
- The OS maintains a waiting line or queue of processes for every one of these services.
- At some point in time, the process will be in a queue waiting for CPU service. At some other point in time, the process will be in a different queue waiting for I/O service.
- When the process completes all service requests, it terminates and exits the system.

Process Management - Review

- Process management is one of the major functions of the operating system; it involves creating processes and controlling their execution.
- In most operating systems, several processes are stored in memory at the same time and the operating system (OS) manages the sharing of the CPU and other resources among the various processes.
- This technique in the operating system is called multiprogramming.

System with Multiples Stations



Context Switch

- When the CPU switches to another process, the OS must save the state of the current process and load the saved state of the new process
- Context switch time is overhead time
- Time is dependent on hardware support

OS Features Needed for Multiprogramming

- I/O routines supplied by the system
- Memory management
- CPU scheduling
- Allocation of devices

Service Demand

- The total CPU and I/O service demands of a process is divided into shorter CPU and I/O requests
- A process will require several CPU and I/O bursts.
- Each CPU request is called a CPU burst
- Each I/O request is called an I/O burst
- In normal processing of a process, it alternates from a CPU burst to an I/O burst and repeats

CPU Service

- The total CPU service demand for a process is the sum of all its CPU bursts
- Each CPU burst has a different duration τ
- The total CPU service requested by process P_i is:

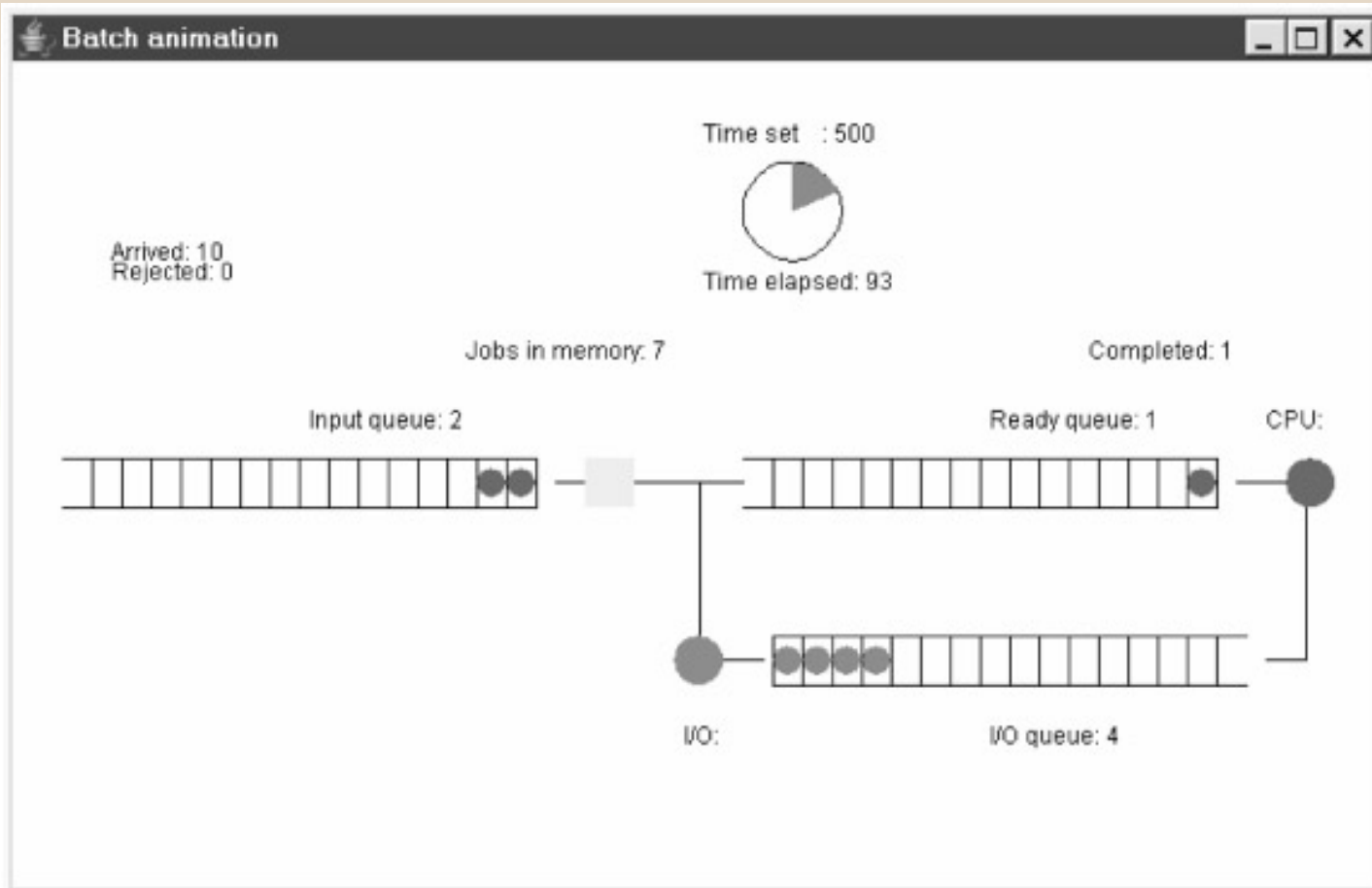
$$\tau_i = \tau_1 + \tau_2 + \dots + \tau_m$$

I/O Demand

- The total I/O service demand for a process is the sum of all its I/O bursts
- Each I/O burst has a different duration, δ
- The total I/O service requested by process P_i is:

$$\delta_i = \delta_1 + \delta_2 + \dots + \delta_n$$

Batch OS with I/O



Occurrence of a Context Switch

- The running process completes its CPU burst.
- The running process is interrupted by a timer under OS control. The process returns to the ready queue.
- A high priority process arrives or is made to transition to the ready state, the OS interrupts the running process if it has a lower priority.

Model of a System with Multiprogramming

- The model of a batch system studied here includes one CPU server and one I/O server.
- Three queues are used:
 - the first queue is used for processes that have arrived and are waiting for memory allocation
 - The ready queue used for the CPU server
 - The I/O queue for the I/O server.
- Each server provides service independently of the other servers (overlapping)

Components of a Simulation Model

- The active resources of the system are modeled as active objects
- The environment is modeled as an active object because the environment generates the jobs that will arrive into the system/model
- The processes are also modeled as active objects --- they arrive requesting resources and services
- The other system resources are modeled as passive objects (memory, queues, etc)

Classes Defined in the Models

- A class for active objects is a class that inherits class Process, which is a Psim library class
- A class for passive objects can be any other class

Classes and Objects in the Simulation Model

- The processes, each one representing a computational unit to be serviced in the computer system, under control of the operating system. The processes are implemented as active objects of class *Job*.
- The environment, which generates the arriving processes. This is an active object of class *Arrivals*. This object creates instances of class *Job* according to the inter-arrival period.
- *The CPU*, which represents the processor that provides CPU service (execution) to the processes according to their service demands. The CPU is an active object of class *Processor*.

Components in the Simulation Model (Cont.)

- *The disk device*, which represents an I/O device that provides I/O service to the processes according to their demands. The disk device is an active object of class *Disk*.
- *The memory*, which represents a major (passive) resource of the system and is modeled as a passive object, of class *Res*.
- *The queues*, these are modeled as passive objects of class *Queue*.

Processes and Multiprogramming

- Processes are program instances
- Several processes can be active at a time (O.S with multiprogramming)
- Only one process is actually running at any instant of time
- The CPU switches from one process to another rapidly (context switching)
- The selection of which process to run next is made by the scheduler

Random Variables

- Random variables are used in the model to represent several parameters and each random variable is derived from a specific probability distribution.
- The inter-arrival periods are generated from an exponential distribution.
- The CPU and I/O service periods, also known as service demands, each generated from an exponential distribution.
- The memory demands for the processes are generated from a uniform distribution.

First Part of Output of A Simulation Run

Psim/Java project: System with Multiprogramming - CPU and I/O

Simulation date: 2/18/2005 17:34

Ave. (mean) inter-arrival per.: 2.3

mean CPU service per.: 18.5; mean I/O service per.: 18.75

Min memory req.: 10, Max memory req: 65

Total system memory: 512

Degree of multiprogramming: 20; Queue size: 125

Job1 requiring service 5.456 arrives at time 0.721

Processor starting CPU burst of Job1 at 0.721

Processor completed CPU burst of Job1 at 2.752

I/O dev starting I/O burst of Job1 at 2.752

I/O dev completed burst Job1 at 4.811

Processor starting CPU burst of Job1 at 4.811

Processor completed CPU burst of Job1 at 6.092

Summary of Output

Simulation closing at: 811.1911086822296
Memory usage: 0.9150302684027927
avg num items used: 476.90258114299934

End Simulation of System with Multiprogramming -
CPU and I/O servers, clock: 811.1911086822296
Results of simulation:

Service factor: 0.048
Total number of jobs that arrived: 231
Total number of rejected jobs: 56
Throughput: 44
Maximum number of jobs in memory: 20
Proportion rejected/arrived jobs: 0.242
Proportion of rejected/completed jobs: 1.273
Average job wait period: 177.092
Processor utilization: 0.969
I/O dev utilization: 0.935

A System with no Multiprogramming

- The C++ simulation model for the batch operating system with I/O is implemented in file batchmio.cpp
- Only one process is allowed to be stored in memory at a time. The degree of multiprogramming is set to 1
- All processes request a certain amount of memory a (passive resource), a CPU burst, an I/O burst, and finally, another CPU burst.