SISTEMAS OPERACIONAIS

Processos: Estruturas de Controle

Processos e Recursos (1)

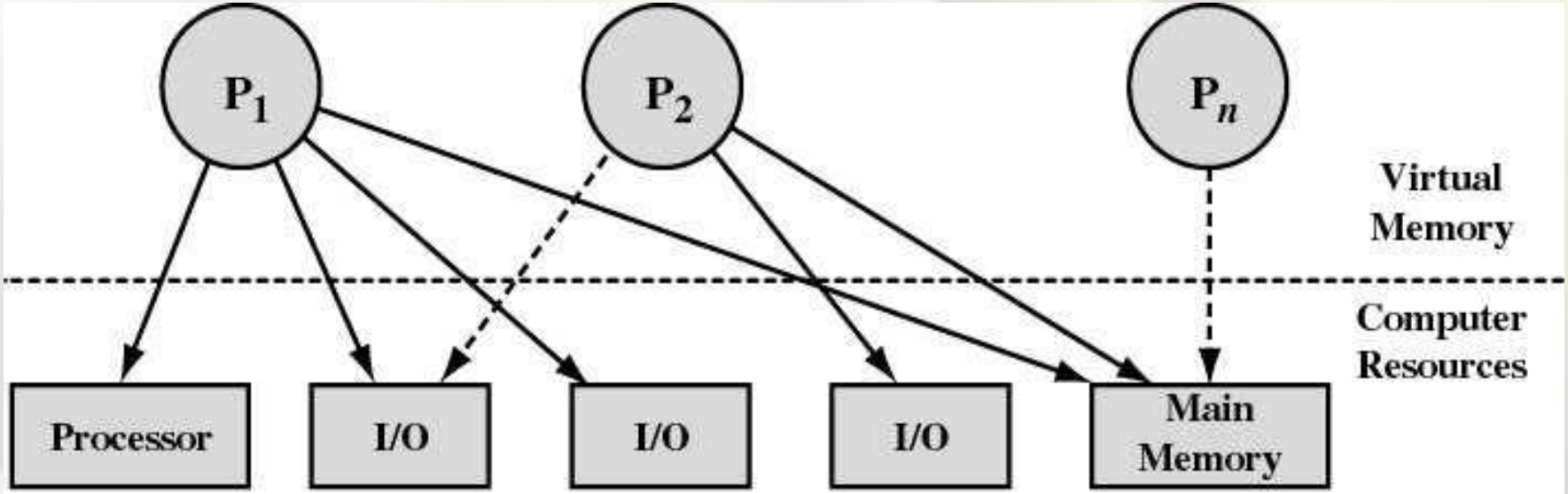


Figure 3.9 Processes and Resources (resource allocation at one snapshot in time)

Processos e Recursos (2)

- O S.O. gerencia recursos computacionais em benefício dos diversos processos que executam no sistema.
- A questão fundamental é:
  - Que **informações** o sistema operacional precisa manter para poder **controlar** os processos e **gerenciar** os recursos em benefícios deles?

Tabelas de Controle do S.O.

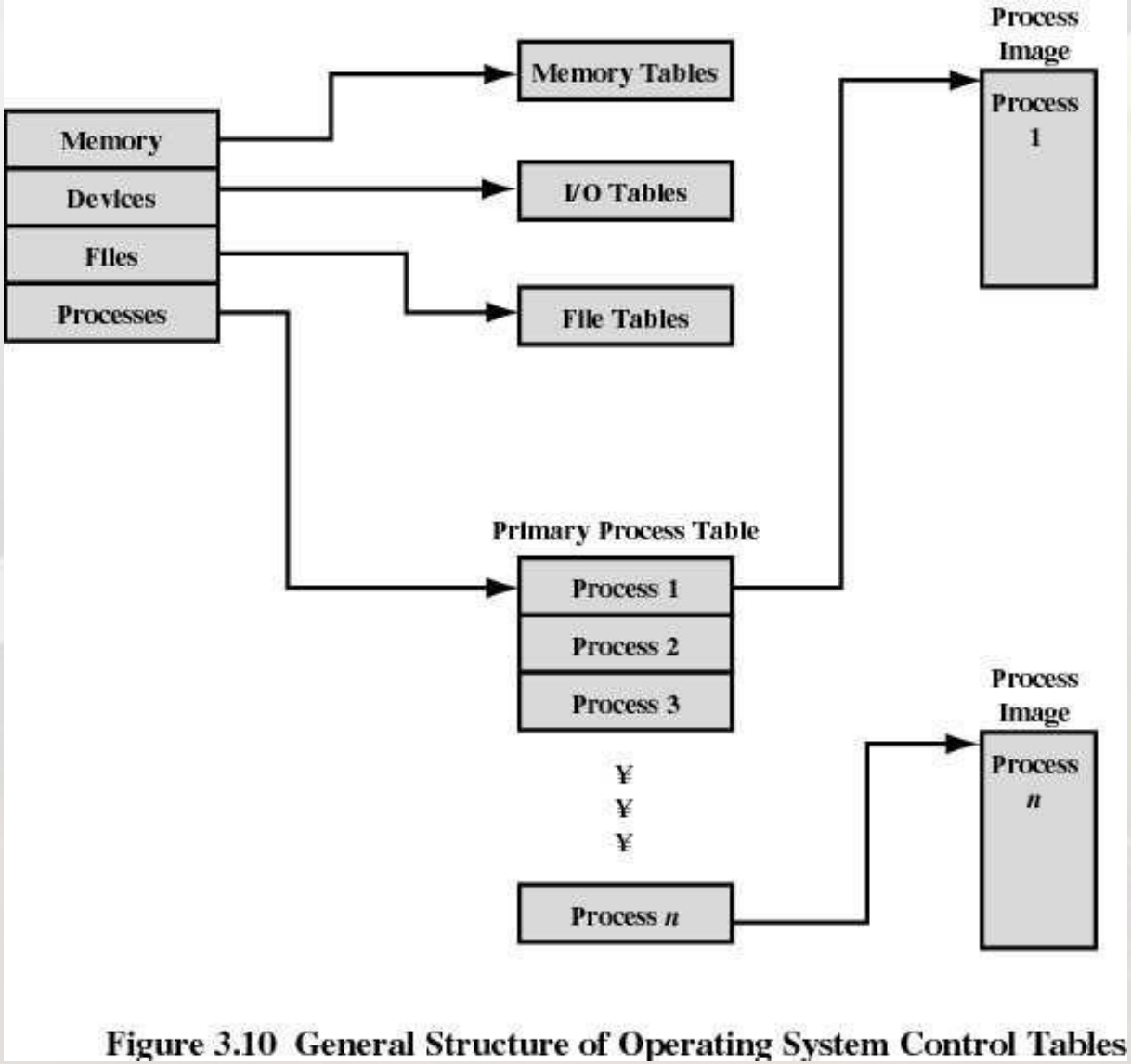
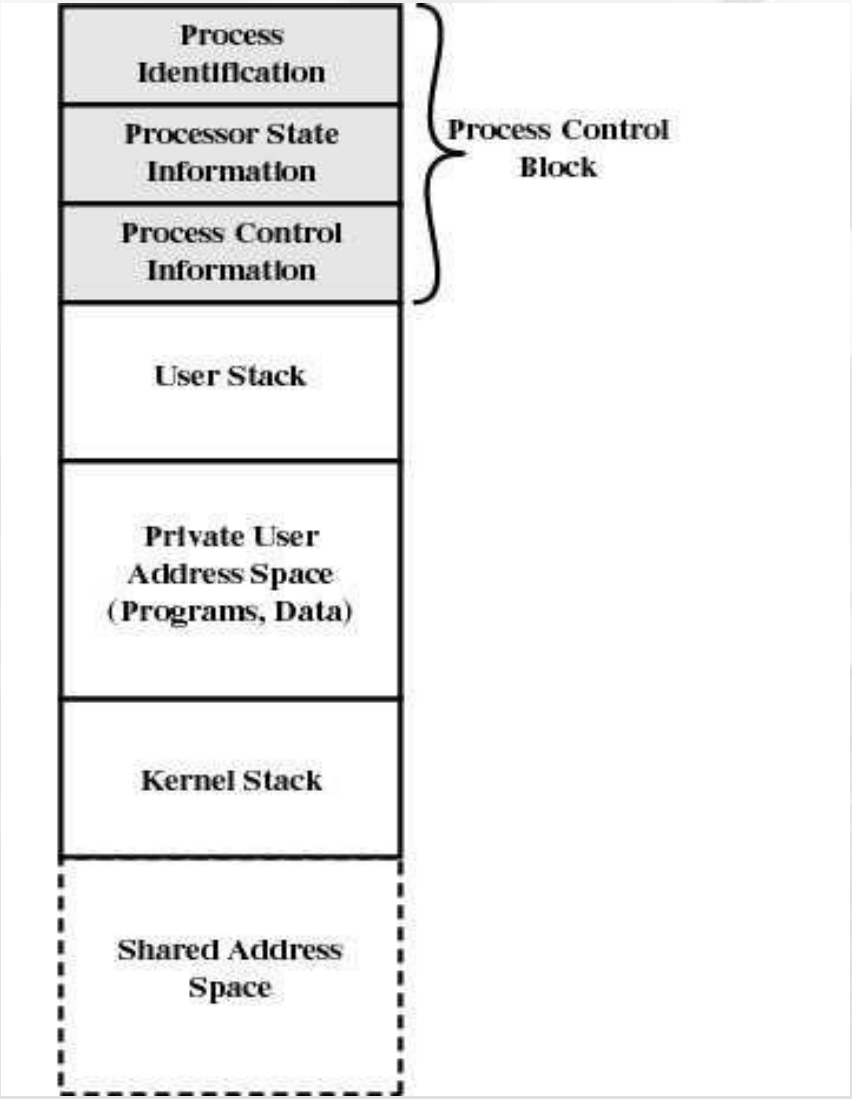


Figure 3.10 General Structure of Operating System Control Tables

Imagem do Processo



- Nome dado à coleção formada por:
  - Código do programa a ser executado.
  - Pilha do sistema para controle de chamadas de procedimentos e de SVCs.
  - Área de dados para armazenamento de variáveis locais e globais.
  - Coleção de atributos do processo

Bloco de Controle de Processos

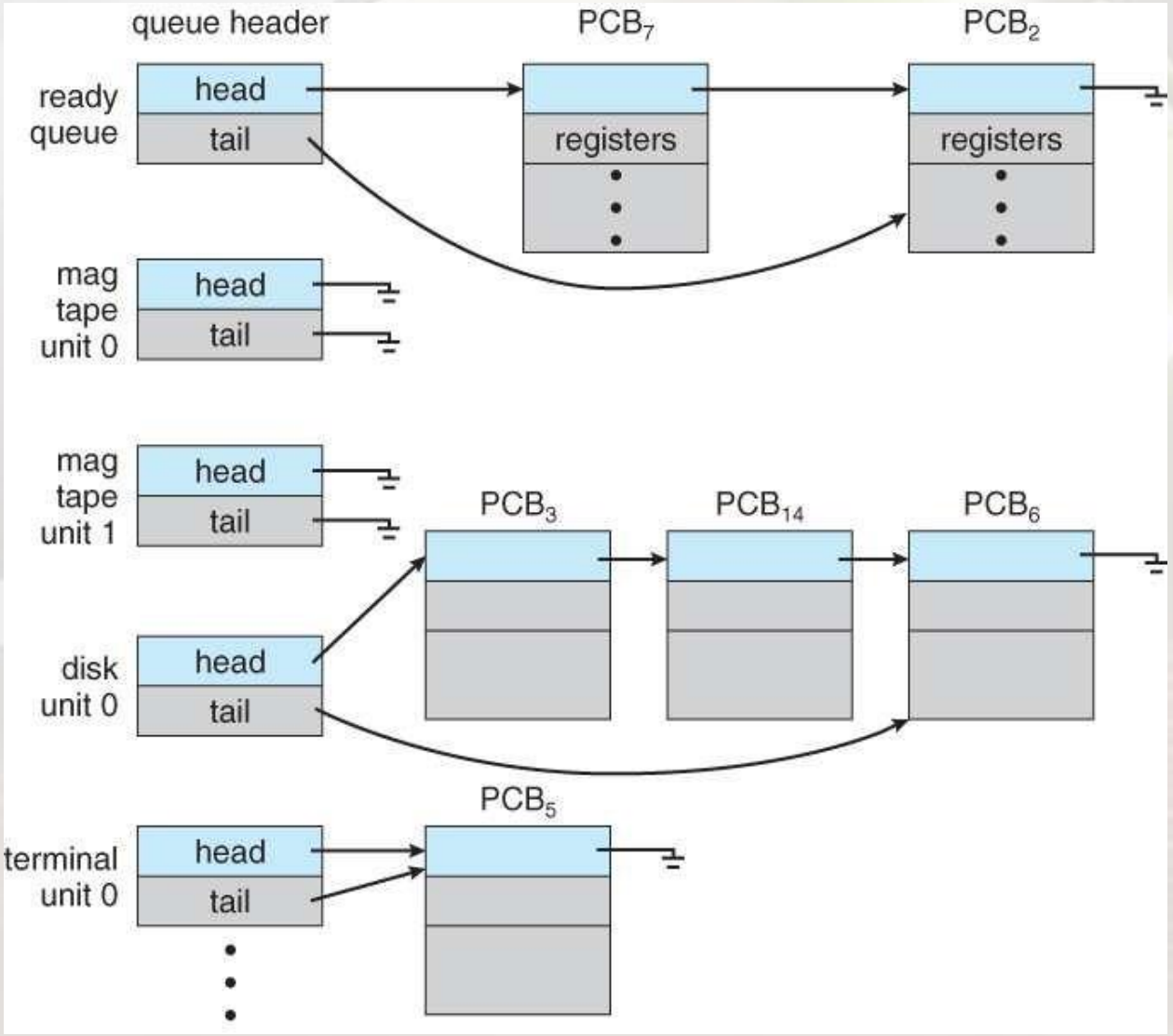
ou

Process Control Block (PCB)

Bloco de Controle de Processo

- Estrutura de dados (registro) usada para representar um processo dentro do S.O.
  - Todas as informações que o S.O. precisa para poder controlar a execução do processo (atributos do processo)
- Número **fixo** ou **variável** de blocos descritores de processos (alocação estática x alocação dinâmica de memória)
- Informações Típicas do BCP
  - Prioridade do processo
  - localização na memória principal
  - Estado do processo
  - Contexto de execução (conteúdo dos registradores)
  - Accounting (ex: uso de CPU)
  - Ponteiros para encadeamento nas filas

PCBs e as Filas do Sistema

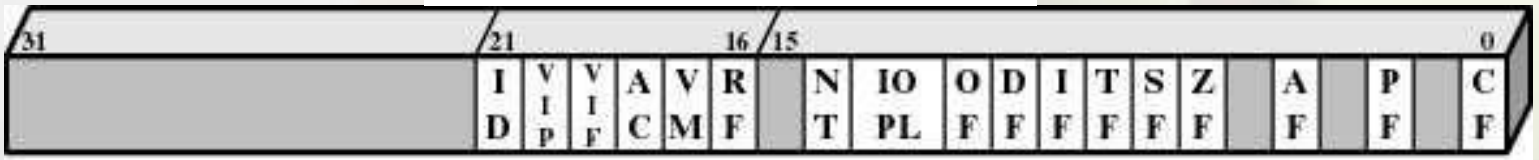


Tipos de Informações do PCB

- As informações mantidas no PCB podem ser divididas em três categorias:
  - Identificação do processo
    - ID do processo, do processo pai, do usuário...
  - Informações de estado do processador
    - Contexto de execução :
      - Registradores visíveis ao usuário
      - Reg. de controle/estado: PC, SP, Flags, Status (modo supervisor /usuário, interrupção habilitada /disabilitada)...
  - Informações de controle do processo
  - ...

Ex: Resgistrador flags

Pentium II EFLAGS Register



ID = Identification flag
VIP = Virtual interrupt pending
VIF = Virtual interrupt flag
AC = Alignment check
VM = Virtual 8086 mode
RF = Resume flag
NT = Nested task flag
IOPL = I/O privilege level
OF = Overflow flag

DF = Direction flag
IF = Interrupt enable flag
TF = Trap flag
SF = Sign flag
ZF = Zero flag
AF = Auxiliary carry flag
PF = Parity flag
CF = Carry flag

AA-64 RFLAGS

	RFLAGS																														
Bits	63..32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13..12	11	10	9	8	7	6	5	4	3	2	1	0
Drapeaux	-	-	-	-	-	-	-	-	-	-	-	ID	VIP	VIF	AC	VM	RF	0	NT	IOPL	OF	DF	IF	TF	SF	ZF	0	AF	0	PF	1	CF

Informações de Controle do Processo (1)

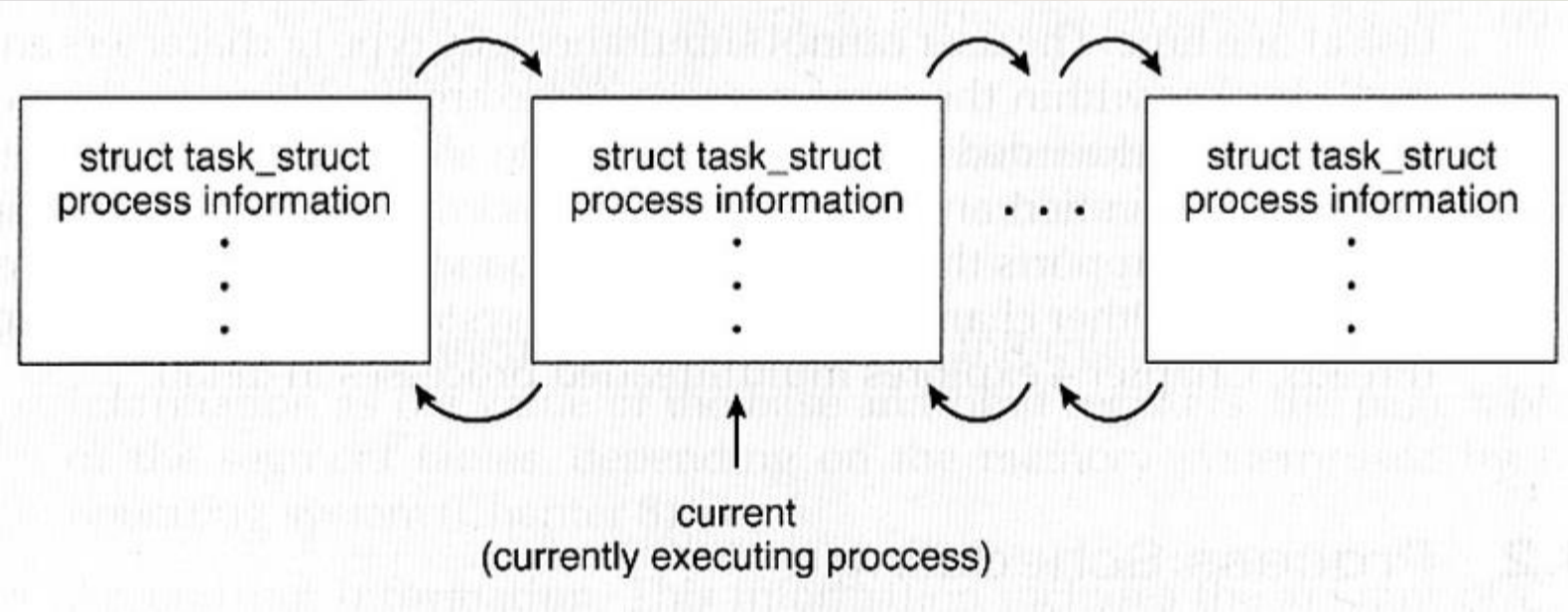
- Informações de Escalonamento e Estado:
  - Estado do processo (ready, running, suspended, etc.)
  - Prioridade (default, corrente, máxima)
  - Tempo de espera na fila
  - Tempo de execução na última fatia de tempo
  - Evento que o processo está aguardando
- Estruturação de dados (ex: ponteiros)
- Comunicação entre processos:
  - Flags, sinais e mensagens podem estar associados com a comunicação entre dois processos independentes.
  - Algumas ou todas essas informações podem estar mantidas no BCP.

Informações de Controle do Processo (2)

- Privilégios em termos de memória que pode ser acessada, instruções que podem ser executadas, ou mesmo serviços e utilitários do sistema.
- Gerência de Memória:
  - Ponteiros para tabelas de páginas/segmentos que descrevem a memória virtual assinalada ao processo.
- *Ownership* e utilização de recursos:
  - Arquivos abertos;
  - Histórico de uso da UCP ou de outro recurso (para usos do escalonador);

Exemplo... PCB no Linux: Estrutura C *task_struct*

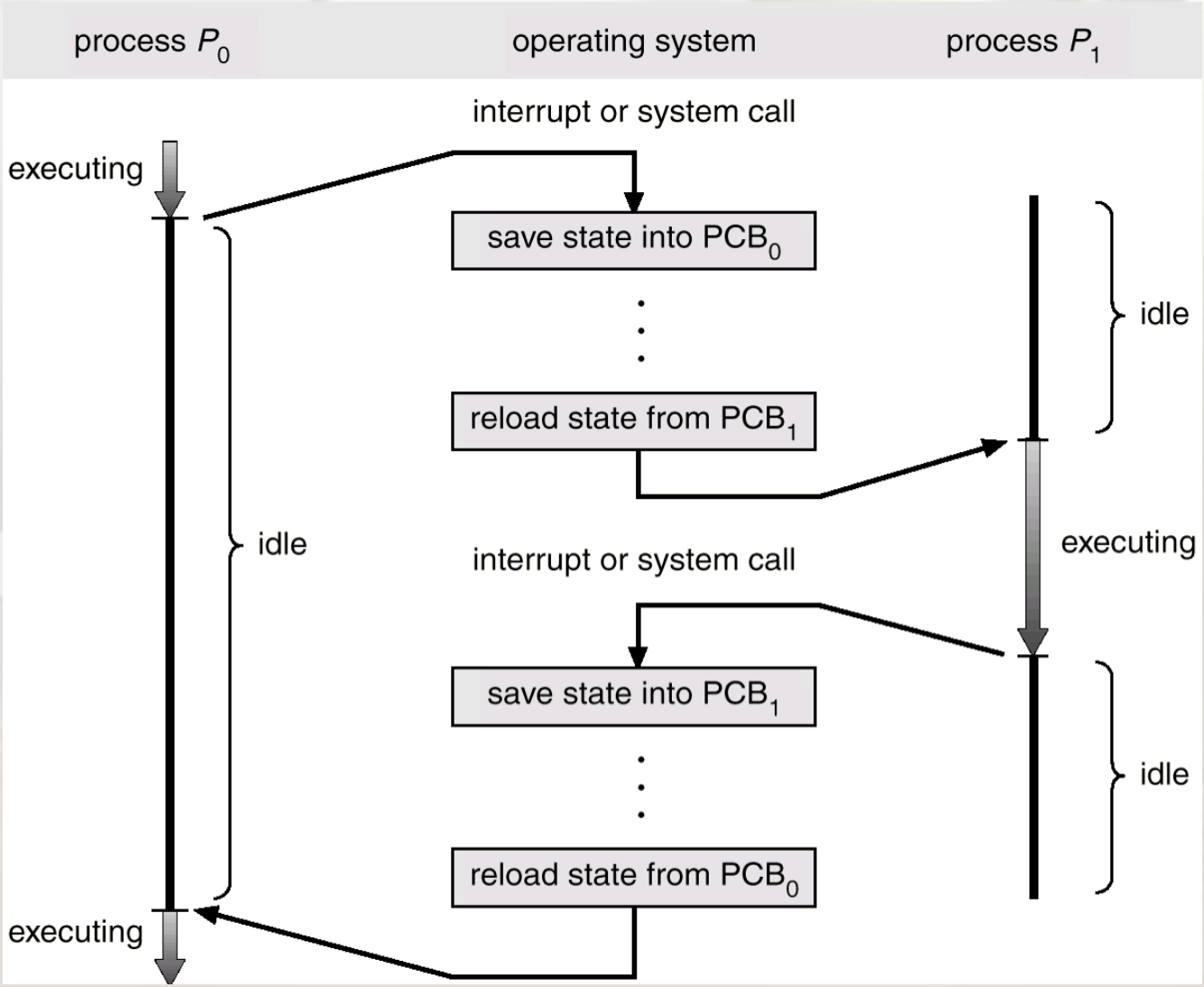
```
pid_t pid; /* process identifier */
long state; /* state of the process */
unsigned int time_slice /* scheduling information */
struct task_struct *parent; /* this process's parent */
struct list_head children; /* this process's children */
struct files_struct *files; /* list of open files */
struct mm_struct *mm; /* address space of this process */
```



Troca de Contexto (1)

- Contexto de execução: estado do processador
- Ações na troca de contexto
  - Salvar o contexto do processador, incluindo o PC e outros registradores.
  - Alterar o PCB do processo que está no estado "em-execução" (*running*).
  - Mover o PCB para a fila apropriada.
  - Selecionar outro processo para execução.
  - Alterar o PCB do processo selecionado.
  - Alterar as tabelas de gerência de memória.
  - Restaurar o contexto do processo selecionado.

Ações na Troca de Contexto (2)



Ações na Troca de Contexto (3)

- Setjmp() and longjmp() are subroutines that let you perform complex flow-of-control in C/Unix.
  - What setjmp() does is save the contents of the registers so that longjmp() can restore them later.
  - Longjmp() resets the registers to the values saved in env. This includes the *sp*, *fp* and *pc*.
    - What this means is that longjmp() doesn't return.

```
#include <stdio.h>
#include <stdlib.h>
#include <setjmp.h>

main()
{
  jmp_buf env;
  int i;

  i = setjmp(env);
  printf("i = %d\n", i);

  if (i != 0)
    exit(0);

  longjmp(env, 2);
  printf("Does this line get
printed?\n");
}
```

http://web.eecs.utk.edu/~plank/plank/classes/cs360/360/notes/Setjmp/lecture.html

O Escalonador ("*Scheduler*")

- Módulo do S.O. responsável pelo controle do recurso "processador".
- Divide o tempo da UCP entre os processos do sistema.
- Três tipos básicos:
  - Escalonador de curto prazo ("*short-term scheduler*");
  - Escalonador de longo prazo ("*long-term scheduler*");
  - Escalonador de médio prazo ("*medium-term scheduler*").

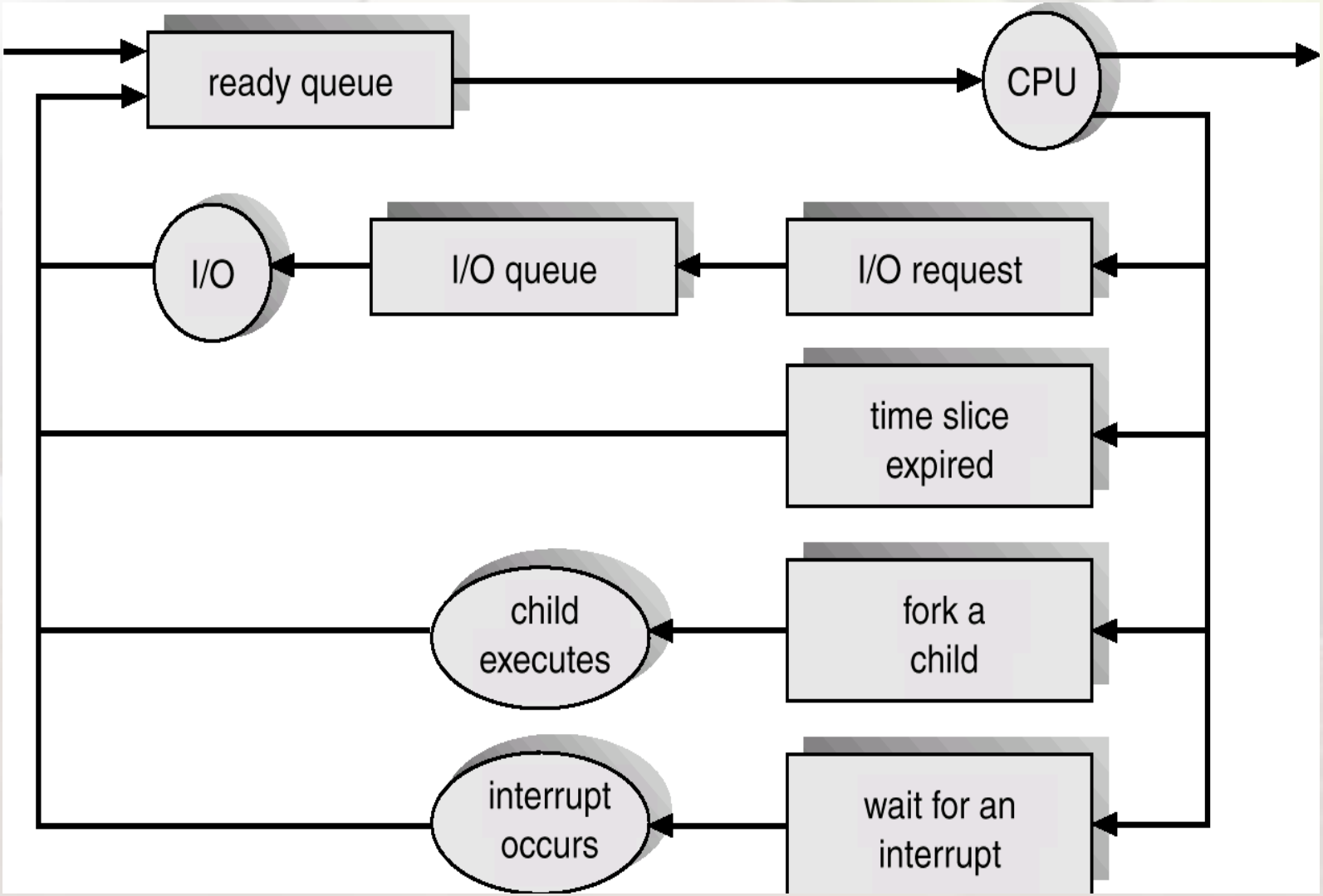
Escalonador de Curto Prazo (1)

- Escalonador da UCP
  - Dispatcher, CPU Scheduler
- Seleciona qual processo deve ser executado a seguir (ready ->running).
- É invocado muito frequentemente (ordem de milisegundos).
  - Deve, portanto, ser rápido.

Escalonador de Curto Prazo (2)

```
#include <iostream>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <stdlib.h>
#include <sched.h>
#include <main.h>

using namespace std;

int main(void)
{
  int pid;
  timespec time;

  // Gets the round robin scheduler time slice interval
  // You can read about this by typing "man 2 sched_rr_get_interval"
  int result = sched_rr_get_interval(0,&time);

  cout << "Result was: " << result << endl;
  cout << "Time (seconds): " << time.tv_sec << endl;
  cout << "Time (nanoseconds): " << time.tv_nsec << endl;

  return 1;
}
```

Escalonador de Curto Prazo (3)



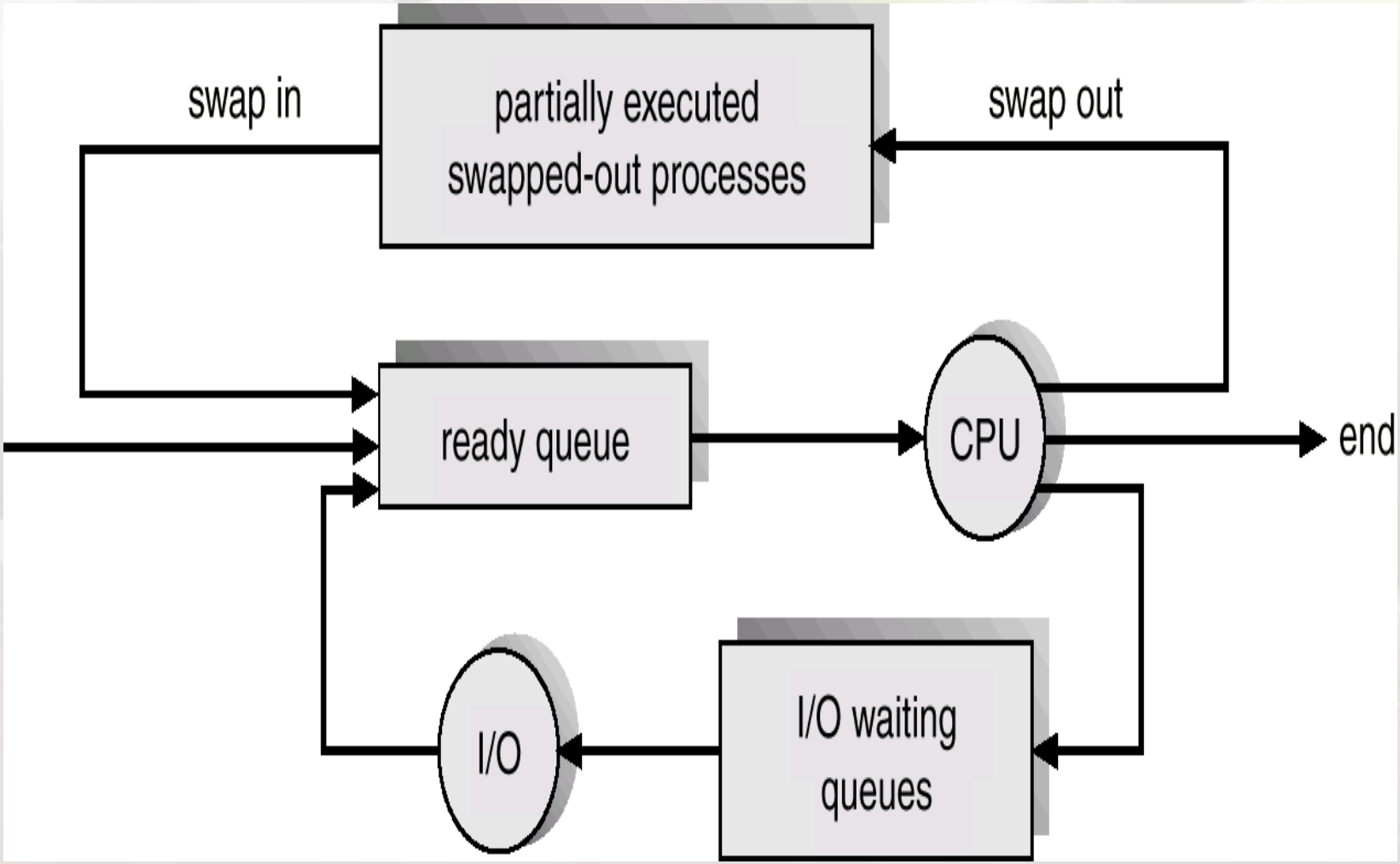
Escalonador de Longo Prazo

- Escalonador de Jobs (“Job Scheduler”).
- Seleciona quais processos devem ser levados para a fila de prontos (new->ready).
- Baixa frequência de invocação (ordem de segundos ou minutos).
- Permite o controle da carga no sistema, (controla o grau de multiprogramação).

Escalonador de Médio Prazo (1)

- Utiliza a técnica de swapping.
  - Swap out: a execução do processo é suspensa e o seu código e dados são temporariamente copiados para o disco.
  - Swap in: o processo é copiado de volta do disco para a memória e sua execução é retomada do ponto onde parou.
- Está intimamente ligado à gerência de memória.

Escalonador de Médio Prazo (2)



Escalonamento e a Transição de Estados

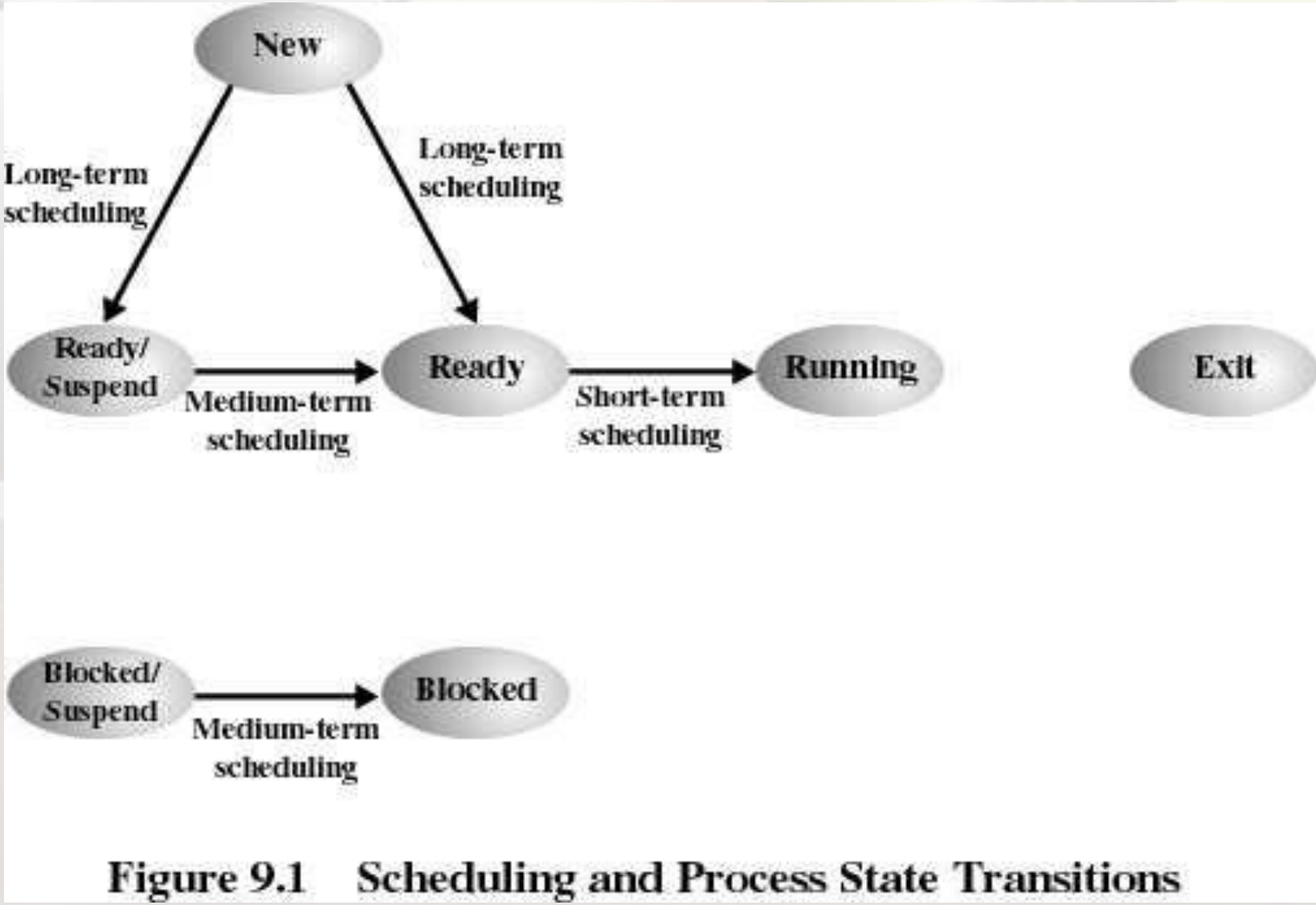


Figure 9.1 Scheduling and Process State Transitions

Escalonamento e as Filas do Sistema

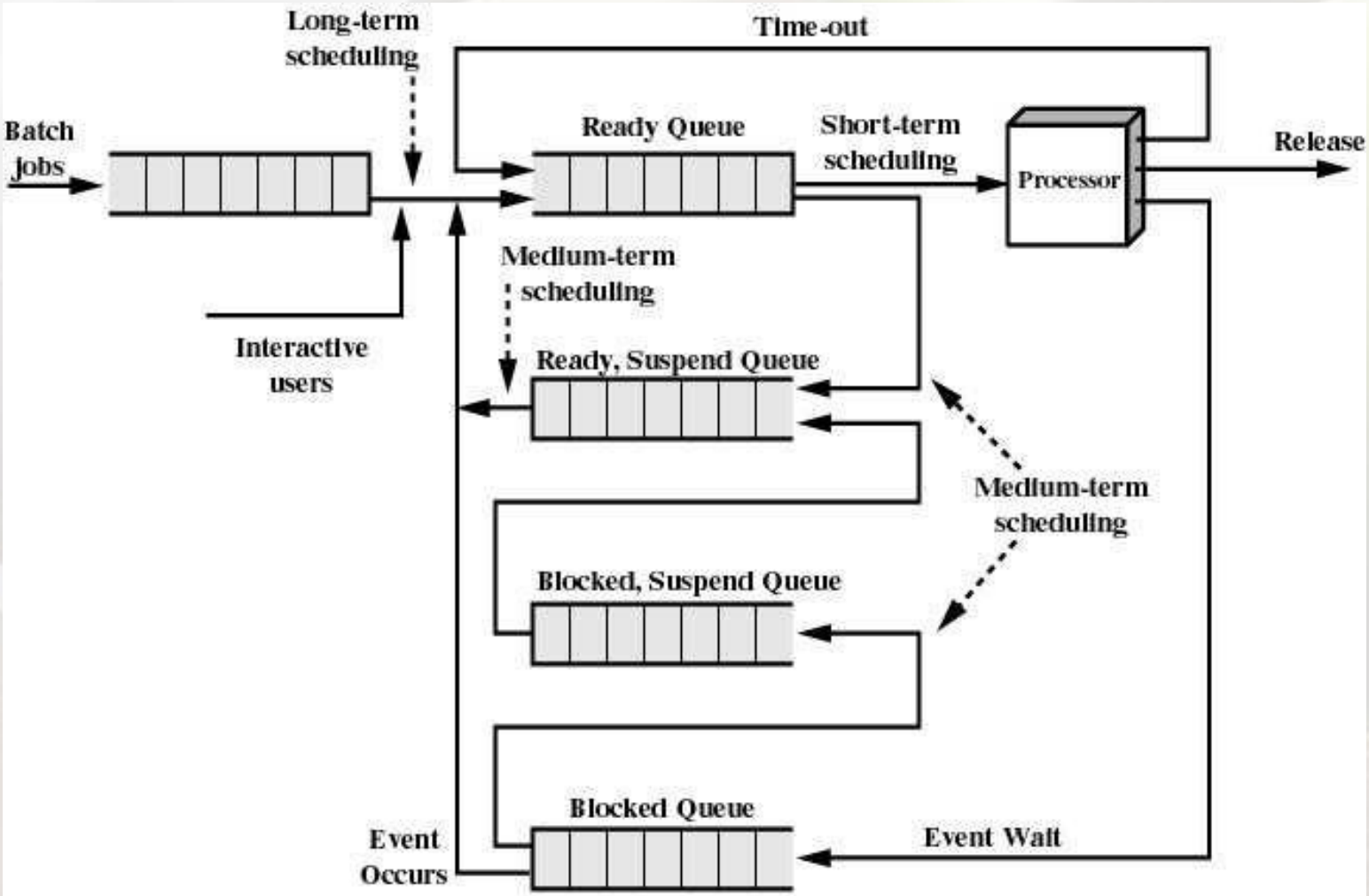
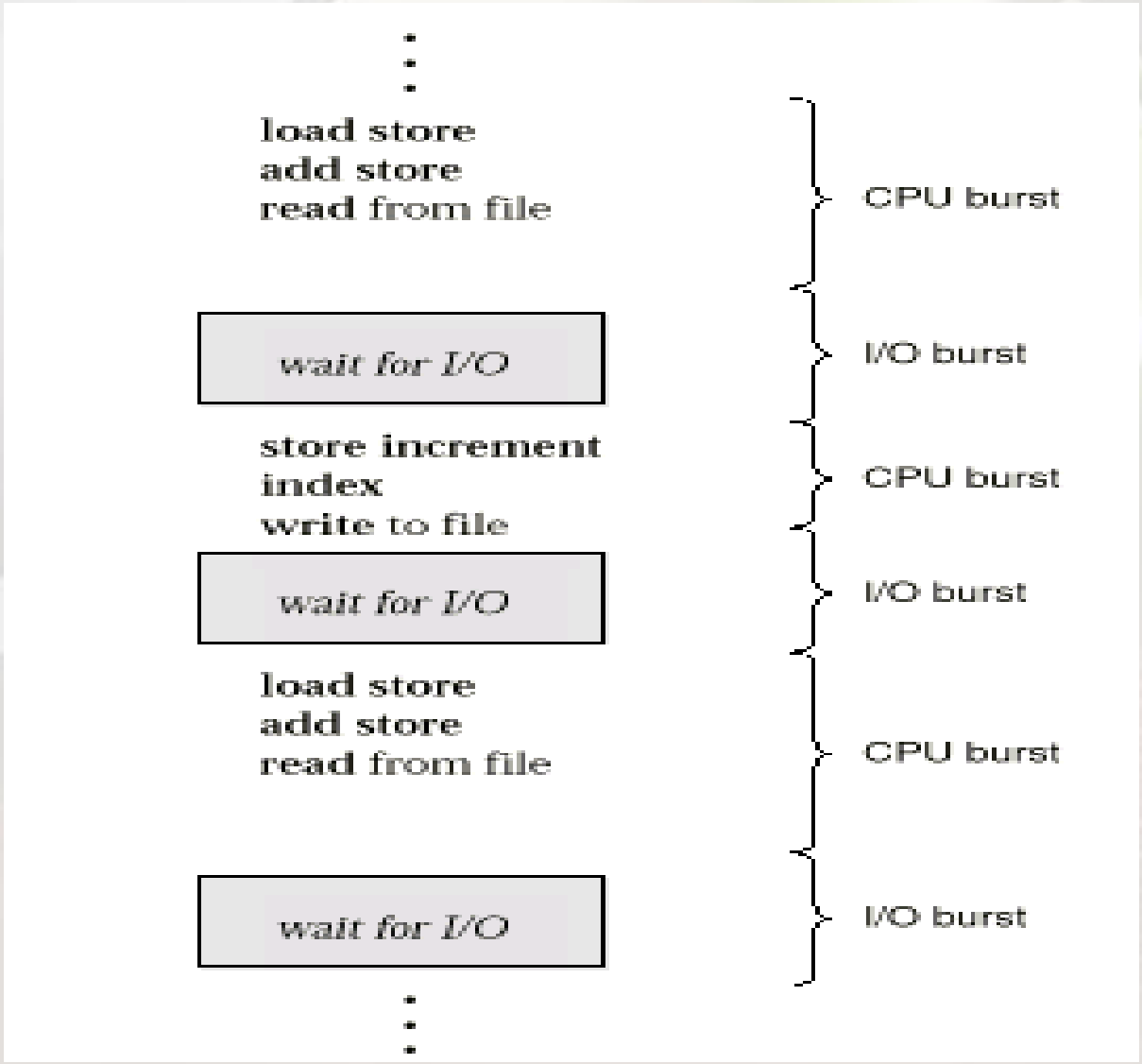


Figure 9.3 Queuing Diagram for Scheduling

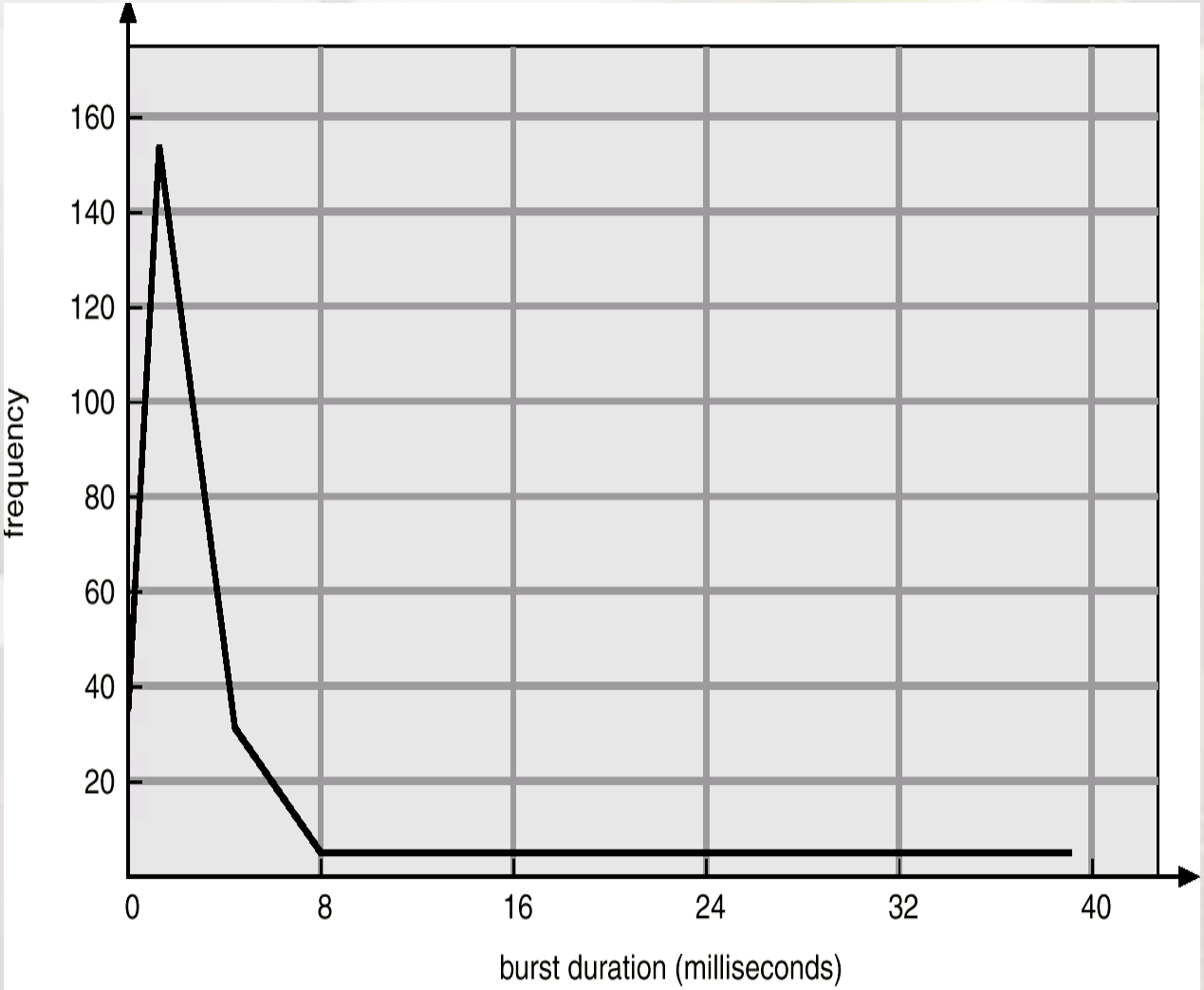
Tipos de Escalonadores (Resumo)

Long-term scheduling	The decision to add to the pool of processes to be executed
Medium-term scheduling	The decision to add to the number of processes that are partially or fully in main memory
Short-term scheduling	The decision as to which available process will be executed by the processor
I/O scheduling	The decision as to which process's pending I/O request shall be handled by an available I/O device

Ciclos de CPU e de I/O (1)



Ciclos de CPU e de I/O (2)



Tipos de Processos

- Processo CPU Bound:
  - Uso intensivo de CPU.
  - Realiza pouca operação de E/S.
  - Pode monopolizar a CPU, dependendo do algoritmo de escalonamento.
- Processo I/O Bound:
  - Orientado a I/O.
  - Devolve deliberadamente o controle da CPU.

Time

- Comando que, além de medir a performance de um processo (tempo) também traz algumas estatísticas sobre o mesmo
  - Modo simples
    - `time <command>`
  - Modo com saída formatada
    - `time -f "Elapsed Time = %E \n Inputs %I" <command>`
    - `time -f "%F %I %K %O %P %W %X %c %k %p %w %x" <command>`
- Help?
  - *man time*
- *No Ubuntu*
  - */usr/bin/time*

- C - Name and command line arguments used
- D - Average size of the process's unshared data area in kilobytes
- E - Elapsed time in a clock format
- F - Number of page faults
- I - Number of file system inputs by the process
- K - Average total memory use of the process in kilobytes
- M - Maximum resident set size of the process during the lifetime in Kilobytes
- O - Number of file system outputs by the process
- P - Percentage of CPU that the job received
- R - Number of minor or recoverable page faults
- S - Total number of CPU seconds used by the system in kernel mode
- U - Total number of CPU seconds used by user mode
- W - Number of times the process was swapped out of main memory
- X - Average amount of shared text in the process
- Z - System's page size in kilobytes
- c - Number of times the process was context switched
- e - Elapsed real time used by the process in seconds
- k - Number of signals delivered to the process
- p - Average unshared stack size of the process in kilobytes
- r - Number of socket messages received by the process
- s - Number of socket messages sent by the process
- t - Average resident set size of the process in kilobytes
- w - Number of time the process was context switched voluntarily
- x - Exit status of the command

Referências

- Slides adaptados de Roberta Lima Gomes (UFES)

- Bibliografia
  - W. Stallings, "Operating Systems: internals and design principles", 6th Edition, Editora Prentice-Hall, 2009.
    - Seções 3.3 e 3.4
  - Silberschatz A. G.; Galvin P. B.; Gagne G.; "Fundamentos de Sistemas Operacionais", 8a. Edição, Editora LTC, 2010.
    - Capítulo 3
  - Deitel H. M.; Deitel P. J.; Choffnes D. R.; "Sistemas Operacionais", 3ª. Edição, Editora Prentice-Hall, 2005
    - Seção 3.3