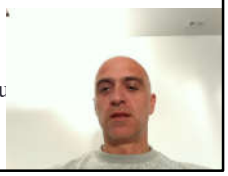
INTRODUÇÃO À GESTÃO DE PROCESSOS

Escola Superior de Tecnologia e Gestão de Beja
Curso de Engenharia Informática
Sistemas Operativos

Lu



Olá! Nestes slides iremos realizar uma introdução à gestão de processos.

Gestão de Processos

- Entrelaçar a execução de vários processos de modo a maximizar a utilização do processador mas mantendo um tempo de resposta aceitável.
- Disponibilizar os recursos necessários aos processos
- Suportar a comunicação entre processos

A gestão de processos tem como objetivo distribuir o processador pelos vários processos, e fornecer a estes os recursos necessários à sua execução.
Também deve disponibilizar mecanismos de comunicação entre processos, para que estes possam colaborar em tarefas conjuntas.

Processo

- Programa a correr em memória

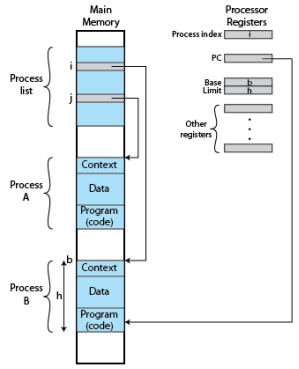


Figure 2.8 Typical Process Implementation

Um processo é um programa a correr em memória.
Cada um dos processos ocupará uma zona da memória, que conterá os seus dados, ou variáveis, e o seu código.
Nesta zona de memória também é armazenado o contexto do programa, ou seja os valores dos registos do computador em utilização pelo programa, para que este possa ser parado em qualquer momento, e retomado mais tarde, na mesma situação
O sistema também necessita de manter uma lista dos processos em execução, para que possa ir atribuindo o processador a cada um destes.

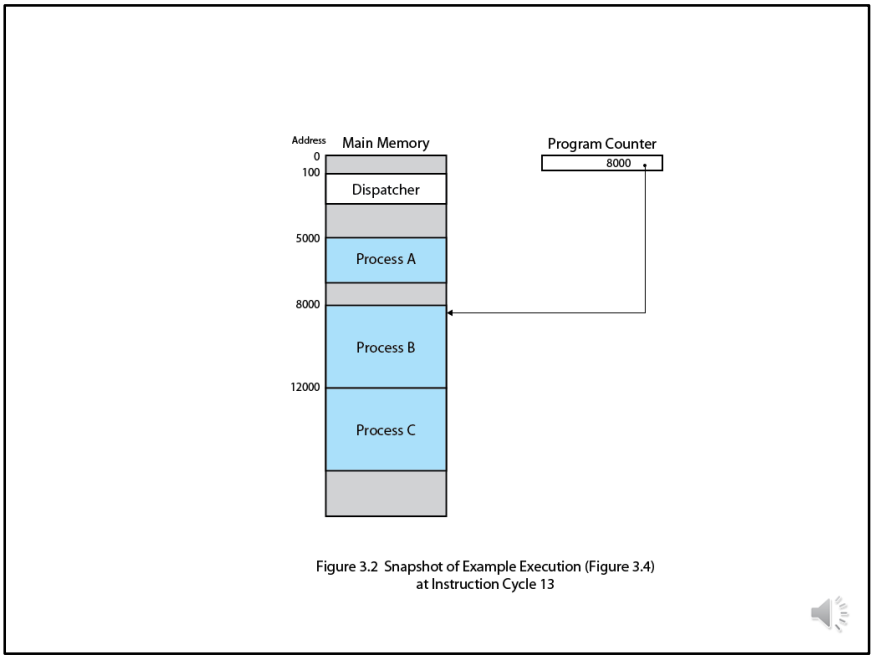


Figure 3.2 Snapshot of Example Execution (Figure 3.4) at Instruction Cycle 13

Nesta figura observamos três processos em memória, e verificamos que o program counter do computador está a apontar para o início do processo B.
Isso indica que neste momento, o sistema irá começar a executar o processo B.

5000	8000	12000
5001	8001	12001
5002	8002	12002
5003	8003	12003
5004		12004
5005		12005
5006		12006
5007		12007
5008		12008
5009		12009
5010		12010
5011		12011
(a) Trace of Process A	**(b) Trace of Process B**	**(c) Trace of Process C**

5000 = Starting address of program of Process A
8000 = Starting address of program of Process B
12000 = Starting address of program of Process C

Figure 3.3 Traces of Processes of Figure 3.2

Nesta figura são apresentados os endereço com as instruções dos três processos.
O processador terá de executar de forma alternada cada um destes processos.

1	5000		
2	5001		
3	5002		
4	5003		
5	5004		
6	5005		
---------- Time out			
7	100		
8	101		
9	102		
10	103		
11	104		
12	105		
13	8000		
14	8001		
15	8002		
16	8003		
---------- I/O request			
17	100		
18	101		
19	102		
20	103		
21	104		
22	105		
23	12000		
24	12001		
25	12002		
26	12003		
		27	12004
		28	12005
		---------- Time out	
		29	100
		30	101
		31	102
		32	103
		33	104
		34	105
		35	5006
		36	5007
		37	5008
		38	5009
		39	5010
		40	5011
		---------- Time out	
		41	100
		42	101
		43	102
		44	103
		45	104
		46	105
		47	12006
		48	12007
		49	12008
		50	12009
		51	12010
		52	12011
		---------- Time out	

100 = Starting address of dispatcher program

shaded areas indicate execution of dispatcher process;
first and third columns count instruction cycles;
second and fourth columns show address of instruction being executed

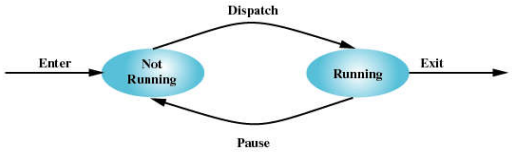
Figure 3.4 Combined Trace of Processes of Figure 3.2

Para conseguir executar os três processos, o sistema começa pelo primeiro a entrar no sistema, o processo A, que começa no endereço 5000 da memória, e executa este processo durante um certo tempo.
Ao fim desse tempo, surge um timeout, que indica que o sistema deve mudar para outro processo.
É selecionado o processo B, pois eventualmente estará há mais tempo à espera de correr. Este processo começa no endereço 8000 da memória.
Mas para realizar esta mudança de processo, têm de ser executadas instruções do próprio sistema, assinaladas a azul na figura.
Após esta troca de processo, o processo B começa a correr, e após quatro instruções realiza um operação de entrada/saída.
Como estas operações demoram tempo a concluir, e não envolvem a utilização do processador, o processador pode ir executar outro processo.
É selecionado o processo C que começa no endereço 12000 da memória. O processo B ficará à espera da conclusão da operação de entrada/saída.
Como verificamos pela figura, foi novamente necessário executar instruções do sistema, a azul, para mudarmos do processo B para o processo C.
A seguir, o sistema continua a distribuir o processador pelos vários processos, correndo um de cada vez, até estes serem concluídos.

Deve notar-se que quando o processador volta a um processo anterior, irá continuar no ponto do programa onde ficou.

Modelo de Processos com 2 Estados

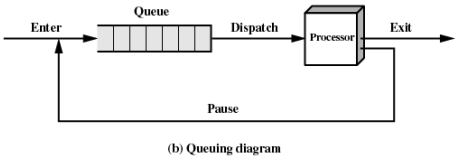
- Um processo pode estar num de dois estados
  - Parado
  - Correndo



(a) State transition diagram

Assim, podemos dizer que um processo pode encontrar-se em dois estados. Parado ou correndo.
Num sistema com um único processador apenas pode estar um processo a correr, enquanto os outros estão parados, à espera de uma fatia de tempo do processador para poder correr.

Processos Parados numa Lista



(b) Queuing diagram

Esta figura apresenta uma lista de espera, onde se encontram todos os processos a aguardar tempo de processador.
Os novos processos, que entram no sistema, são colocados no fim desta fila, e ficam a aguardar tempo de processador.
O programa que esteja a correr pode ter de voltar ao fim da fila de espera, caso o tempo que lhe tenha sido atribuído não seja suficiente para terminar.

Processos Bloqueados

- Parado
  - preparado para execução
- Bloqueado
  - à espera de uma operação de E/S
- O escalonador não pode simplesmente colocar a correr o processo que está há mais tempo à espera pois este pode estar bloqueado

Existem, como vimos anteriormente, alguns processos que não podem avançar na execução do seu programa enquanto não ocorrer um determinado evento, como por exemplo a conclusão de uma operação de entrada/saída.
Estes processos são denominados de bloqueados pois não podem avançar enquanto o evento não ocorrer.
Como não podem avançar no programa, não deve ser selecionados para correr no processador.

Modelo de Processos com 5 Estados

- Correndo
- Preparado
- Bloqueado
- Novo
- Saída

Assim, para uma boa gestão dos processos devem ser considerados estes cinco estados.

Modelo de Processos com 5 Estados

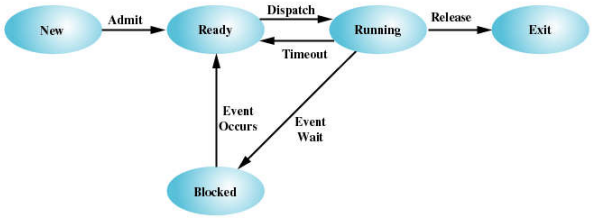


Figure 3.6 Five-State Process Model

No estado *new, ou novo,* os processos são admitidos no sistema, é-lhes atribuída memória, e outros recursos que estes necessitem.
Após a criação do processo, este passa para o estado ready, ou pronto, e poderá ser selecionado a qualquer momento para correr.
Quando um processo é selecionado para correr passa para o estado running, em execução ou correndo, e mantêm esse estado durante um curto espaço de tempo.
Se o processo esgotar a sua parcela de tempo, então terá de voltar a esperar por nova fatia de tempo, ficando no estado ready.
Se estiver a correr e realizar uma operação de entrada/saída então irá para o estado blocked, ou bloqueado, até que essa operação termine.
Quando essa operação terminar voltará para o estado ready.
Se o processo estiver a correr, e terminar, então entrará no estado exit, onde serão destruídos todos os recursos que lhe estavam atribuídos.

Estados dos Processos

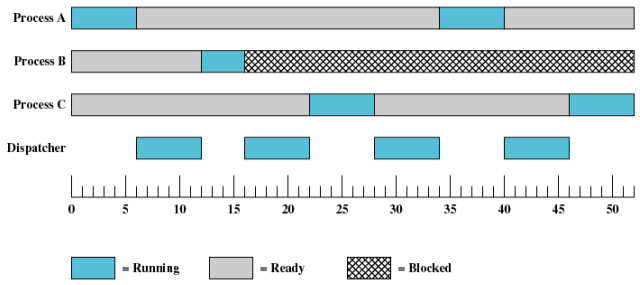
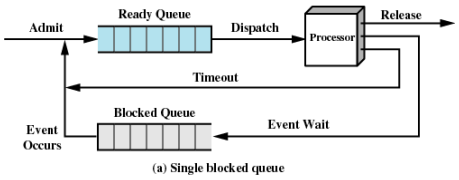


Figure 3.7 Process States for Trace of Figure 3.4

Nesta página podemos observar os estados dos processos A, B, e C, já apresentados anteriormente.
Verificamos que em cada momento apenas um processo se encontra no estado running, assinalado com a cor azul.
Primeiro é o processo A que corre, enquanto todos os outros estão ready, com a cor cinzenta.
Depois corre o dispatcher do sistema, que realiza a troca de processo, colocando o processo B a correr.
O processo B, depois de correr um pouco, realiza uma operação de entrada/saída, e fica no estado bloqueado, assinalado com o padrão de riscas.
O dispatcher corre novamente para colocar a correr o processo C, e depois vai distribuindo o processador pelo processa A, e processo C.
O processo B mantém-se bloqueado durante a janela de tempo apresentada no diagrama.

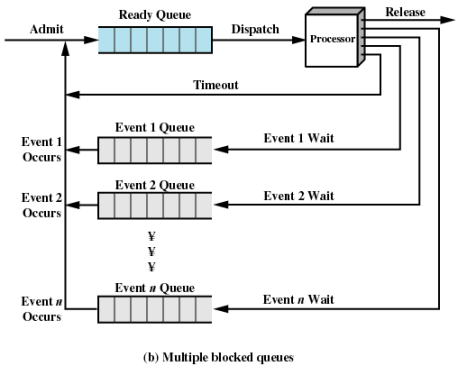
Utilizando Duas Filas



(a) Single blocked queue

Nesta figura incorpora-se na gestão dos processos, uma fila de processos bloqueados, que permite colocar em espera os processos que aguardam um determinado evento, como por exemplo a conclusão de uma operação de entrada/saída.
Enquanto permanecerem nesta fila, os processos nunca serão selecionados para correr no processador.
Quando o evento ocorre, por exemplo a conclusão da operação de entrada/saída, o processo é colocado na fila ready, e logo que seja selecionado poderá voltar a correr.

Várias Filas



(b) Multiple blocked queues

Figure 3.8 Queuing Model for Figure 3.6

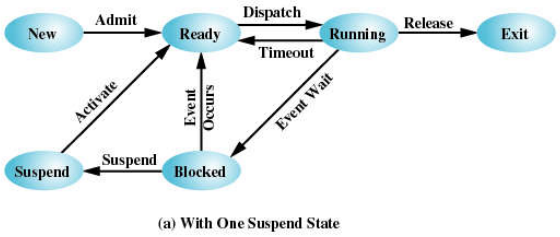
Para uma melhor gestão dos processos bloqueados, estes poderão estar distribuídos por diferentes filas de espera, cada qual associada a um determinado tipo de evento, como por exemplo operações de entradas/saídas no disco, ou operações de entradas/saídas em equipamentos de comunicação.

Processos Suspensos

- O processador é mais rápido que as E/S e por isso todos os processos podem estar à espera de E/S
- Estes processos podem ser colocados em disco para libertarem memória (*Swapping*)
- O estado Bloqueado torna-se no estado Suspenso quando os processo são colocados em disco

Em determinadas situações, quando a memória RAM está muito ocupada, e é necessário executar novos programas, pode ser preciso colocar processos em disco. Os melhores candidatos a ir para disco são os processos bloqueados pois como estão parados, poderão aguardar pelos eventos que os bloquearam, em disco.
Desta forma minimiza-se o impacto desta operação no desempenho do sistema.
Quando um processo bloqueado é colocado em disco fica no estado suspenso.

Estado de Suspensão



(a) With One Suspend State

Nesta figura acrescenta-se o estado suspenso ao diagrama de estados dos processos. Descrevemos apenas a parte nova do diagrama, que está relacionada com os processos suspensos.
Se for necessário libertar espaço em memória, o sistema pode passar processos bloqueados para disco, ficando estes no estado suspenso.
Se entretanto ocorrer o evento que um destes processos aguarda, então o processo passará logo para RAM e será colocado no estado ready, para que possa correr logo que selecionado.

Troca de Processo (1)

- Interrupção do Clock
  - o processo foi executado ininterruptamente durante o intervalo de tempo máximo permitido
- Interrupção de E/S
- Chamada ao sistema
  - como por exemplo a abertura de um ficheiro

Como vimos, para que seja possível executar vários processos num computador, é necessário trocar constantemente, o processo que está a correr no processador.
Esta troca pode ser provocada por diversas situações.
O mais normal é o processo estar a correr, e esgotar o tempo que lhe tinha sido atribuído.
O sistema operativo é informado que este tempo terminou através de uma interrupção do clock do sistema.
Nesse momento o processo que estava a correr é retirado do processador, e colocado na fila ready.
A seguir é escolhido desta lista, outro pocesso para correr.
Por outro lado, se surgir uma operação de entrada/saída isso significa que um dos processos bloqueados pode retomar a sua execução. Se for um processo prioritário poderá ser colocado a correr.
Finalmente, se for realizada uma chamada ao sistema, dependendo do serviço solicitado, o sistema operativo poderá optar por colocar outro processo a correr a seguir.
No caso de ser uma operação de entrada/saída tal terá de acontecer.

Troca de Processo (2)

- Falha de Memória
  - O endereço de memória encontra-se no disco e por isso deve ser colocado em memória principal
- Trap
  - ocorreu um erro
  - pode provocar a passagem do processo para o estado Saída

Em situações de falha de memória, ou erro, também pode existir uma troca de processo.
Se por exemplo faltar uma página a um processo, será gerada uma interrupção de falha de memória, e o sistema tratará dessa interrupção lendo essa página do disco para memória.
Mas enquanto isso acontece deve ser colocado outro processo a correr, para se aproveitar o tempo de processado, e por isso tem de ser realizada uma troca de processo.
Um exemplo de erro pode ser um acesso ilegal a memória, ou seja o acesso a uma zona de memória fora do processo, e isso provocará uma interrupção – a trap, ou armadilha.
O sistema tratará desta trap terminando o processo que provocou o erro, e colocará outro processo ready a correr.

Estado do Processador

- Conteúdo dos registos do processador
  - Registos visíveis ao utilizador
  - Registos de controlo de estado
  - Apontadores de stack
- Program status word (PSW)
  - Contém informação de estado
  - Exemplo: O registo EFLAGS nos processadores Pentium

Sempre que há uma troca de processo, é necessário guardar o estado do processador, ou seja o conteúdo de todos os seus registos, para que o processo possa voltar a correr mais tarde, a partir do mesmo ponto em que se encontrava.
Se isso não acontecer, os próximos processos a correr alterarão os valores dos registos do processador, e nada estará igual, quando pretendermos correr novamente o processo que saiu do processador.
Por isso, em cada troca de processo, é necessário guardar o estado do processador para o processo que é parado, e reposto o estado do processador, para o processo que vai voltar a correr.

Registo EFLAGS do Pentium II

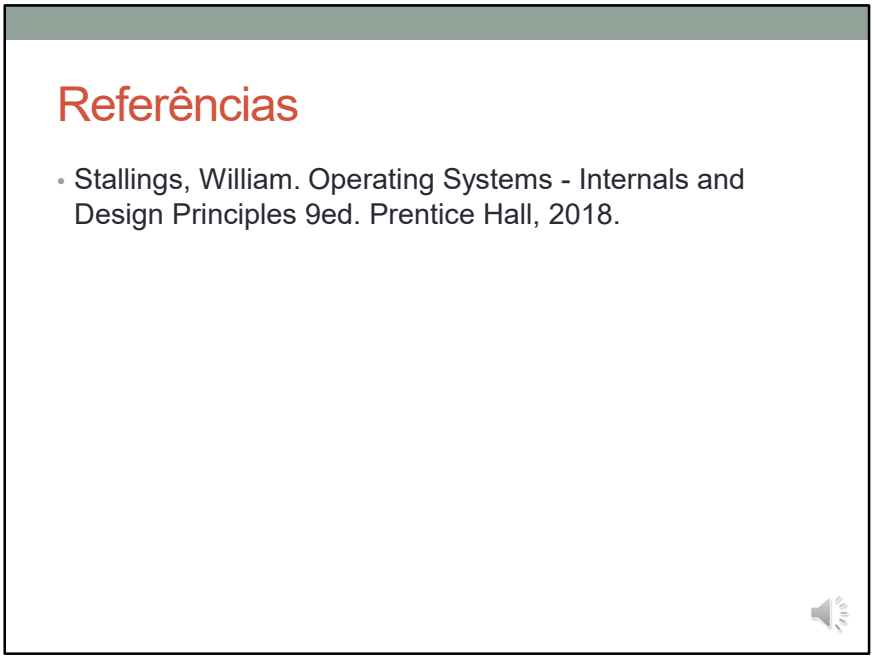
ID	=	Identification flag	DF	=	Direction flag
VIP	=	Virtual interrupt pending	IF	=	Interrupt enable flag
VIF	=	Virtual interrupt flag	TF	=	Trap flag
AC	=	Alignment check	SF	=	Sign flag
VM	=	Virtual 8086 mode	ZF	=	Zero flag
RF	=	Resume flag	AF	=	Auxiliary carry flag
NT	=	Nested task flag	PF	=	Parity flag
IOPL	=	I/O privilege level	CF	=	Carry flag
OF	=	Overflow flag			

Apresentamos nesta figura um exemplo de um registo de um processador onde são armazenadas diversas informações sobre a operação em curso.
Como este, existem outros registos com informações importantes, que deverão ser guardadas sempre que existe uma troca de processo.

Troca de Processo (3)

- Salvaguarda do contexto do processador, incluindo o program counter e outros registos
- Selecção de outro processo para execução
- Restauração do contexto do processo seleccionado

Assim, antes de trocarmos de processo, é necessário guardar o contexto do processador, ou seja o estado do processador, incluindo o program counter e outros registos.
Depois deve ser selecionado um processo ready para execução.
E antes deste novo processo começar a correr devemos restaurar o seu contexto, para que este retome a sua execução desde o ponto em que ficou, na última vez que correu.



Fica aqui referência utilizada para a construção destes slides.