

SO

Sistemas Operacionais

Prof. Rodrigo Martins
rodrigo.martins@francomontoro.com.br



Cronograma

- Processos e seus “primos”, os threads.
- Exercícios

Processos

- Processos são uma das mais antigas e importantes abstrações que os sistemas operacionais proporcionam.
- Eles dão suporte à possibilidade de haver operações (pseudo) concorrentes mesmo quando há apenas uma CPU disponível, transformando uma única CPU em múltiplas CPUs virtuais.

Processos

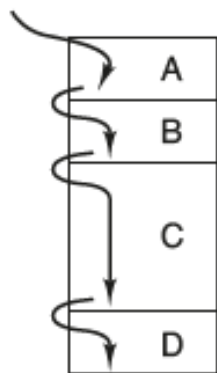
- Base para implementação de sistemas multiprogramáveis.
- A gerencia é função do SO, que para executar os programas é associado um programa a um processo.
- Ao executar um programa o usuário tem impressão de possuir todo processador para seu uso. Não é verdade, visto que os recursos estão sendo compartilhados e a unidade de processamento também.

Processos

- Todos os softwares executáveis no computador, às vezes incluindo o sistema operacional, são organizados em uma série de processos sequenciais, ou, simplesmente, **processos**.

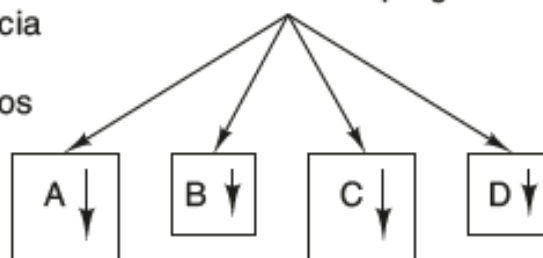
(a) Multiprogramação de quatro programas. (b) Modelo conceitual de quatro processos sequenciais independentes. (c) Apenas um programa está ativo de cada vez.

Um contador de programa

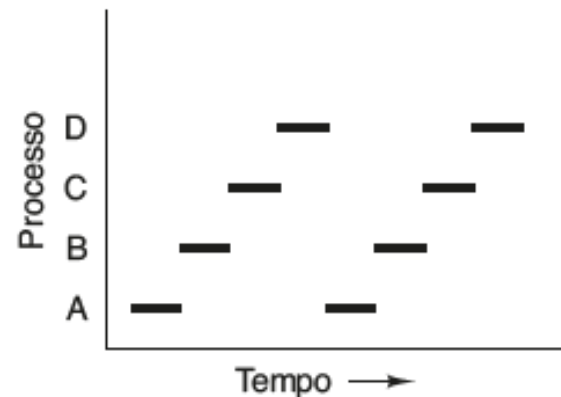


(a)

Quatro contadores de programa



(b)



(c)

Processos

- Considere um cientista de computação que gosta de cozinhar e está preparando um bolo de aniversário para sua filha mais nova. Ele tem uma receita de um bolo de aniversário e uma cozinha bem estocada com todas os ingredientes: farinha, ovos, açúcar, extrato de baunilha etc. Nessa analogia, a receita é o programa, isto é, o algoritmo expresso em uma notação adequada, o cientista de computação é o processador (CPU) e os ingredientes do bolo são os dados de entrada. O processo é a atividade consistindo na leitura da receita, busca de ingredientes e preparo do bolo por nosso cientista.

Processos

- Agora imagine que o filho do cientista de computação aparece correndo chorando, dizendo que foi picado por uma abelha. O cientista de computação registra onde ele estava na receita (o estado do processo atual é salvo), pega um livro de primeiros socorros e começa a seguir as orientações. Aqui vemos o processador sendo trocado de um processo (preparo do bolo) para um processo mais prioritário (prestar cuidado médico), cada um tendo um programa diferente (receita versus livro de primeiros socorros). Quando a picada de abelha tiver sido cuidada, o cientista de computação volta para o seu bolo, continuando do ponto onde ele havia parado.

Criação de Processos

- Quatro eventos principais fazem com que os processos sejam criados:
 1. Inicialização do sistema.
 2. Execução de uma chamada de sistema de criação de processo por um processo em execução.
 3. Solicitação de um usuário para criar um novo processo.
 4. Início de uma tarefa em lote.

Criação de Processos

- Quando um sistema operacional é inicializado, em geral uma série de processos é criada. Alguns desses processos são de primeiro plano, isto é, processos que interagem com usuários (humanos) e realizam trabalho para eles.
- Outros operam no segundo plano e não estão associados com usuários em particular, mas em vez disso têm alguma função específica. Por exemplo, um processo de segundo plano pode ser projetado para aceitar e-mails, ficando inativo a maior parte do dia, mas subitamente entrando em ação quando chega um e-mail.
- Processos que ficam em segundo plano para lidar com algumas atividades, como e-mail, páginas da web, notícias, impressão e assim por diante, são chamados de daemons.

Término de Processos

- Após um processo ter sido criado, ele começa a ser executado e realiza qualquer que seja o seu trabalho. No entanto, nada dura para sempre, nem mesmo os processos. Cedo ou tarde, o novo processo terminará, normalmente devido a uma das condições a seguir:
 1. Saída normal (voluntária).
 2. Erro fatal (involuntário).
 3. Saída por erro (voluntária).
 4. Morto por outro processo (involuntário).

Término de Processos

- A maioria dos processos termina por terem realizado o seu trabalho. Quando um compilador termina de traduzir o programa dado a ele, o compilador executa uma chamada para dizer ao sistema operacional que ele terminou. Essa chamada é exit em Linux e Exit-Process no Windows.
- A segunda razão para o término é a que o processo descobre um erro fatal. Por exemplo, se um usuário digita o comando cc foo.c para compilar o programa foo.c e não existe esse arquivo, o compilador simplesmente anuncia esse fato e termina a execução.

Término de Processos

- A terceira razão para o término é um erro causado pelo processo, muitas vezes decorrente de um erro de programa. Exemplos incluem executar uma instrução ilegal, referenciar uma memória não existente, ou dividir por zero.
- A quarta razão pela qual um processo pode ser finalizado ocorre quando o processo executa uma chamada de sistema dizendo ao sistema operacional para matar outro processo. Em Linux, essa chamada é kill. No Windows é TerminateProcess.

Hierarquia de Processos

- Em alguns sistemas, quando um processo cria outro, o processo pai e o processo filho continuam a ser associados de certas maneiras. O processo filho pode em si criar mais processos, formando uma hierarquia de processos. Observe que um processo tem apenas um pai (mas zero, um, dois ou mais filhos).
- No Linux, um processo e todos os seus filhos e demais descendentes formam juntos um grupo de processos. Quando um usuário envia um sinal do teclado, o sinal é entregue a todos os membros do grupo de processos associados com o teclado no momento.
- Em comparação, o Windows não tem conceito de uma hierarquia de processos. Todos os processos são iguais. O único indício de uma hierarquia ocorre quando um processo é criado e o pai recebe um identificador especial (chamado de handle) que ele pode usar para controlar o filho.

Gerenciador de Tarefas

ArquivoOpçõesExibir

Processos

Desempenho

Histórico de aplicativos

Inicializar

Usuários

Detalhes

Serviços

Nome	Status	2% CPU	64% Memória	5% Disco	0% Rede
Aplicativos (5)					
Windows Explorer		0,1%	39,8 MB	0 MB/s	0 Mbps
Microsoft PowerPoint (32 bits)		0%	196,5 MB	0 MB/s	0 Mbps
Google Chrome		0,4%	224,1 MB	0,1 MB/s	0 Mbps
Gerenciador de Tarefas		0,4%	11,3 MB	0 MB/s	0 Mbps
Adobe Acrobat Reader DC (32 b...		0%	148,3 MB	0 MB/s	0 Mbps
Processos em segundo plano (...)					
xampp-control.exe (32 bits)		0,3%	5,7 MB	0 MB/s	0 Mbps
WMI Provider Host (32 bits)		0%	1,9 MB	0 MB/s	0 Mbps
Windows Setup API		0%	1,6 MB	0 MB/s	0 Mbps
Windows Driver Foundation - Pr...		0%	3,4 MB	0 MB/s	0 Mbps
UninstallerMonitor (32 bits)		0%	11,0 MB	0 MB/s	0 Mbps
Uninstall Programs (32 bits)		0%	1,7 MB	0 MB/s	0 Mbps
Synaptics TouchPad 64-bit Enh...		0%	1,2 MB	0 MB/s	0 Mbps
Synaptics Pointing Device Helper		0%	0,2 MB	0 MB/s	0 Mbps
SQL Server VSS Writer - 64 Bit		0%	1,1 MB	0 MB/s	0 Mbps
SoftThinks Agent Service (32 bits)		0%	2,2 MB	0 MB/s	0 Mbps
Serviço Gateway de Camada de ...		0%	0,7 MB	0 MB/s	0 Mbps
RichVideo Module (32 bits)		0%	1,0 MB	0 MB/s	0 Mbps

Menos detalhes

Finalizar tarefa

Processo e Serviços

- **Processos** são conjuntos de instruções executadas com um determinado propósito. Alguns programas se subdividem em diversos processos.
- Alguns processos não correspondem a programas (“**serviços**”), que rodam em segundo plano para dar suporte ao sistema operacional).

Gerenciador de Tarefas

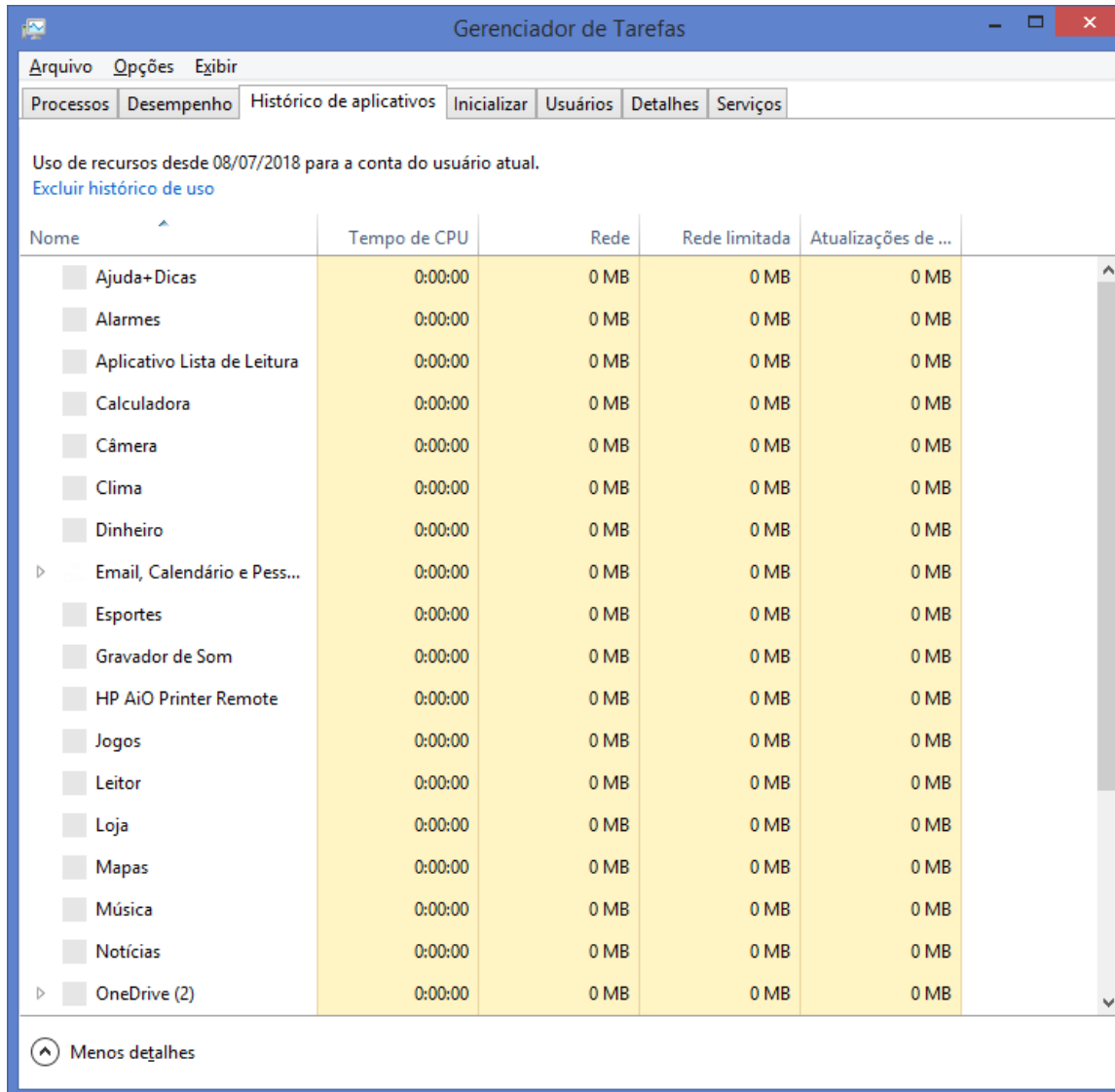
Arquivo Opções Exibir

Processos Desempenho Histórico de aplicativos Inicializar Usuários Detalhes **Serviços**

Nome	PID	Descrição	Status	Grupo
WSearch	7592	Windows Search	Em execução	
WMPNetworkSvc		Serviço de Compartilhamento de Rede do Windo...	Parado	
wmiApSrv	880	Adaptador de Desempenho WMI	Em execução	
WinDefend		Serviço Windows Defender	Parado	
WdNisSvc		Serviço de Inspeção de Rede do Windows Defender	Parado	
wbengine		Serviço de Mecanismo de Backup em Nível de BL...	Parado	
Warsaw Technology	10888	Warsaw Technology	Em execução	
VSSStandardCollectorService...		Visual Studio Standard Collector Service 150	Parado	
VSS		Cópia de Sombra de Volume	Parado	
VsEtwService120		Visual Studio ETW Event Collection Service	Parado	
vds		Disco Virtual	Parado	
VaultSvc	896	Gerenciador de Credenciais	Em execução	
UI0Detect		Deteção de Serviços Interativos	Parado	
TrustedInstaller		Instalador de Módulos do Windows	Parado	
Tomcat8		Apache Tomcat 8.0 Tomcat8	Parado	
SwitchBoard		Adobe SwitchBoard	Parado	
Steam Client Service		Steam Client Service	Parado	
SQLWriter	2860	Gravador VSS do SQL Server	Em execução	
sppsvc		Proteção de Software	Parado	
Spooler	1776	Spooler de Impressão	Em execução	
SNMPTRAP		Interceptação SNMP	Parado	
SftService	5844	SoftThinks Agent Service	Em execução	
SamSs	896	Gerente de Contas de Segurança	Em execução	
RtkAudioService	1228	Realtek Audio Service	Em execução	
RpcLocator	2832	Alocador Remote Procedure Call (RPC)	Em execução	
RichVideo	2768	Cyberlink RichVideo Service(CRVS)	Em execução	
PSL_SVC_2_x64	2780	Corel License Validation Service V2 x64, Powered ...	Em execução	
PerfHost		Host de DLL de Contador de Desempenho	Parado	
ose		Office Source Engine	Parado	

Menos detalhes | Abrir Serviços

Aplicativos

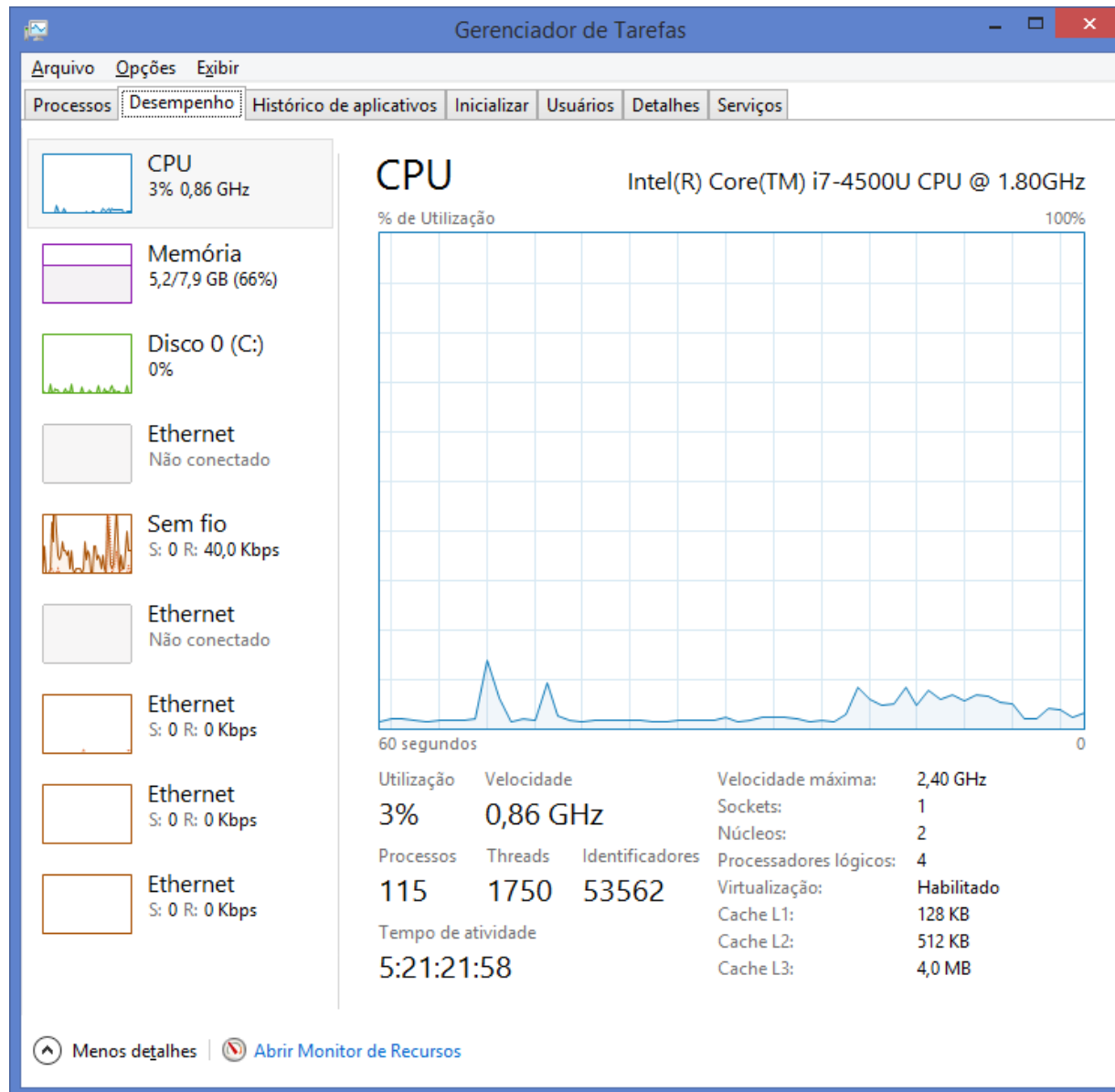


The screenshot shows the Windows Task Manager application with the 'Aplicativos' (Applications) tab selected. The window title is 'Gerenciador de Tarefas'. The menu bar includes 'Arquivo', 'Opções', and 'Exibir'. The tab bar shows 'Processos', 'Desempenho', 'Histórico de aplicativos' (selected), 'Inicializar', 'Usuários', 'Detalhes', and 'Serviços'. Below the tabs, a message states: 'Uso de recursos desde 08/07/2018 para a conta do usuário atual.' with a link 'Excluir histórico de uso'. The main area contains a table with columns: 'Nome', 'Tempo de CPU', 'Rede', 'Rede limitada', and 'Atualizações de ...'. The table lists various applications, all showing 0 MB of network usage. At the bottom, there is a 'Menos detalhes' button with an upward arrow icon.

Nome	Tempo de CPU	Rede	Rede limitada	Atualizações de ...
Ajuda+ Dicas	0:00:00	0 MB	0 MB	0 MB
Alarmes	0:00:00	0 MB	0 MB	0 MB
Aplicativo Lista de Leitura	0:00:00	0 MB	0 MB	0 MB
Calculadora	0:00:00	0 MB	0 MB	0 MB
Câmera	0:00:00	0 MB	0 MB	0 MB
Clima	0:00:00	0 MB	0 MB	0 MB
Dinheiro	0:00:00	0 MB	0 MB	0 MB
▶ Email, Calendário e Pess...	0:00:00	0 MB	0 MB	0 MB
Esportes	0:00:00	0 MB	0 MB	0 MB
Gravador de Som	0:00:00	0 MB	0 MB	0 MB
HP AiO Printer Remote	0:00:00	0 MB	0 MB	0 MB
Jogos	0:00:00	0 MB	0 MB	0 MB
Leitor	0:00:00	0 MB	0 MB	0 MB
Loja	0:00:00	0 MB	0 MB	0 MB
Mapas	0:00:00	0 MB	0 MB	0 MB
Música	0:00:00	0 MB	0 MB	0 MB
Notícias	0:00:00	0 MB	0 MB	0 MB
▶ OneDrive (2)	0:00:00	0 MB	0 MB	0 MB

Menos detalhes

Desempenho...

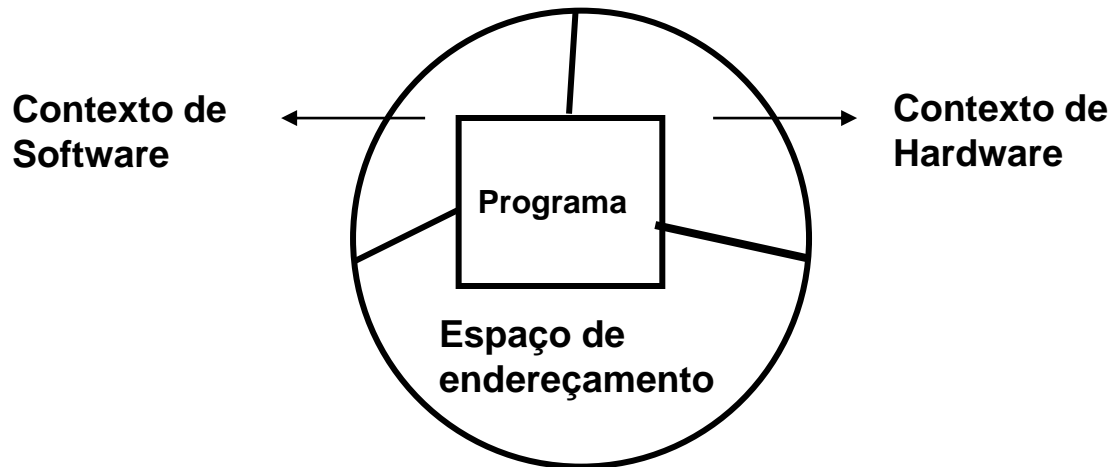


Monitor de Recursos

Essa marcação com uma linha azul mostra a porcentagem da frequência máxima do processador que está em uso no momento. Isso é devido ao sistema de economia de energia do processador, que reduz o clock do mesmo em momentos de pouco uso. No meu caso, como o processador não está sendo muito requisitado, a frequência dele é diminuída para 2% dos 1.8Ghz, ou seja, fica em 0.86Ghz para economizar energia.

Estrutura de um processo

- Como ocorre toda hora uma troca de programas a ser executado é necessário que todas as informações do programa interrompido sejam guardadas para que quando retornar a execução.
- Um processo é formado por três partes.



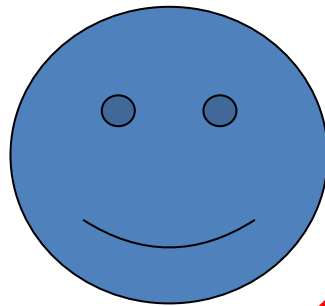
Estrutura do processo

- **Contexto de hardware:** armazena informações da CPU e do programa. Exemplo que endereço da memória esta locado.
- **Contexto de software:** são especificadas características e limites dos recursos que podem ser alocados pelo processo
 - Ex. numero máximo de arquivos abertos, ou qtd máxima de ocupação da memória.
- **Espaço de endereçamento:** instruções e dados do programa, onde parou (ex. linha10).

Estrutura do processo

- Quando o processo esta sendo executado seu contexto de hardware esta armazenando no processador.
- Quando o processo perde o processamento o sistema salva as informações no contexto de hardware do processo.

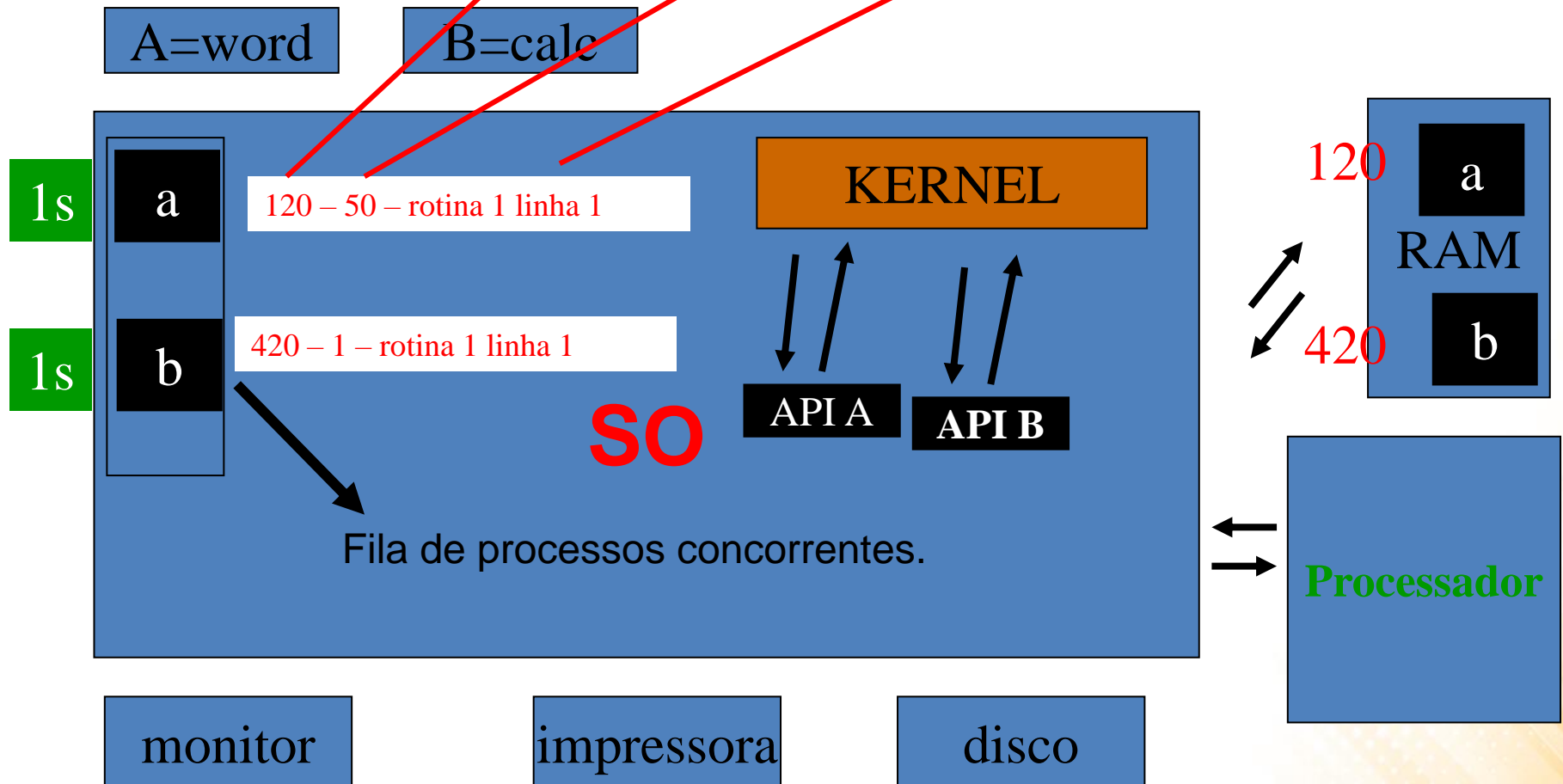
Tempo 0.



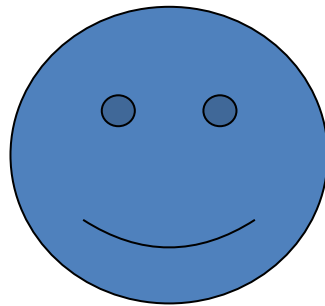
Contexto hardware
Ex. posição memória

Contexto software
Ex. qtd arquivo simultâneos

Endereçamento
Ex. posição execução (em binário)
Porque não tem código fonte).



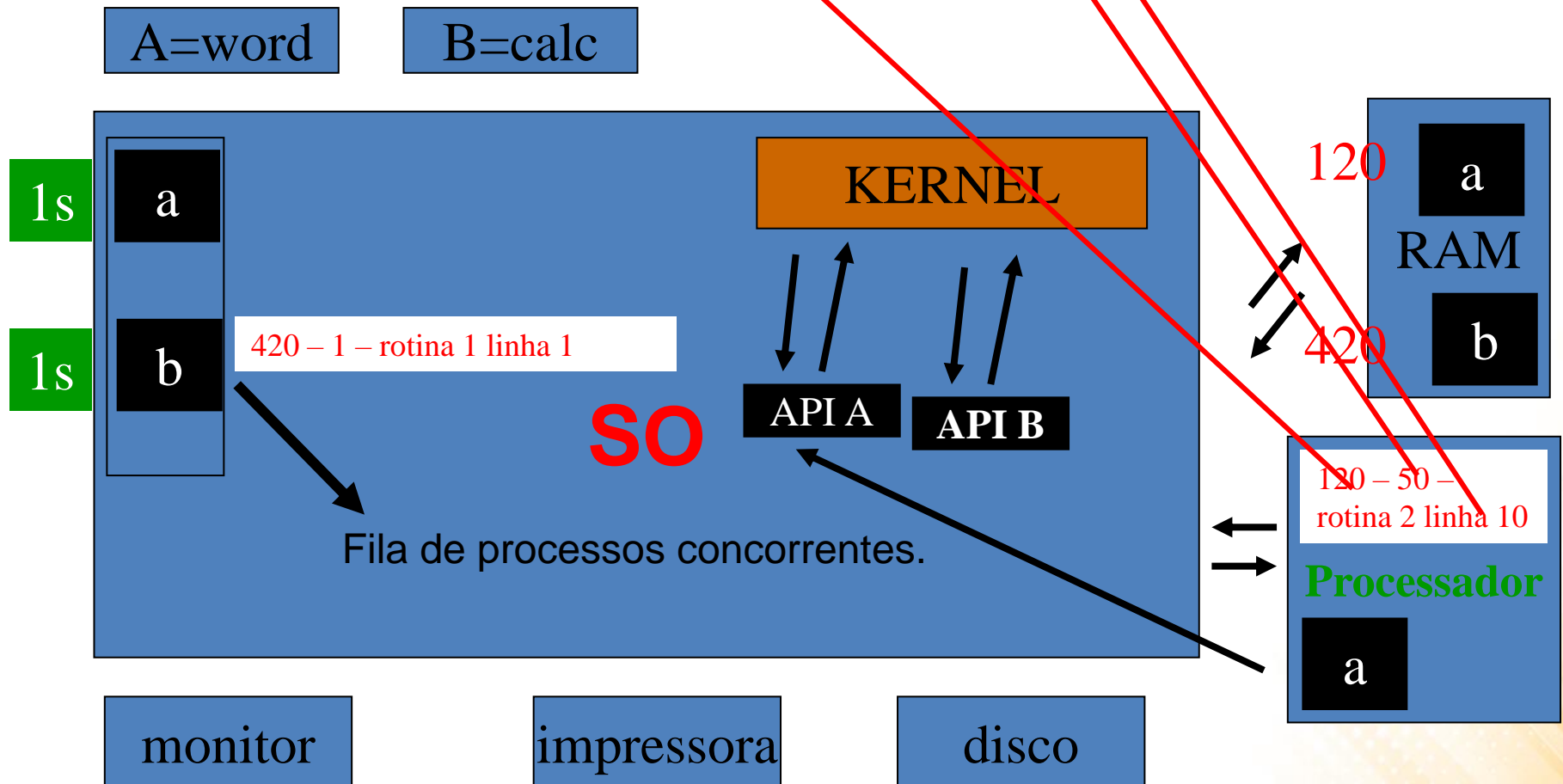
Tempo 1.



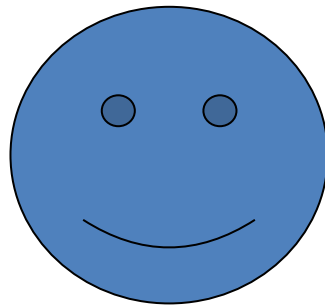
Contexto hardware
Ex. posição memória

Contexto software
Ex. qtd arquivo simultâneos

Endereçamento
Ex. posição execução (em binário)
Porque não tem código fonte).



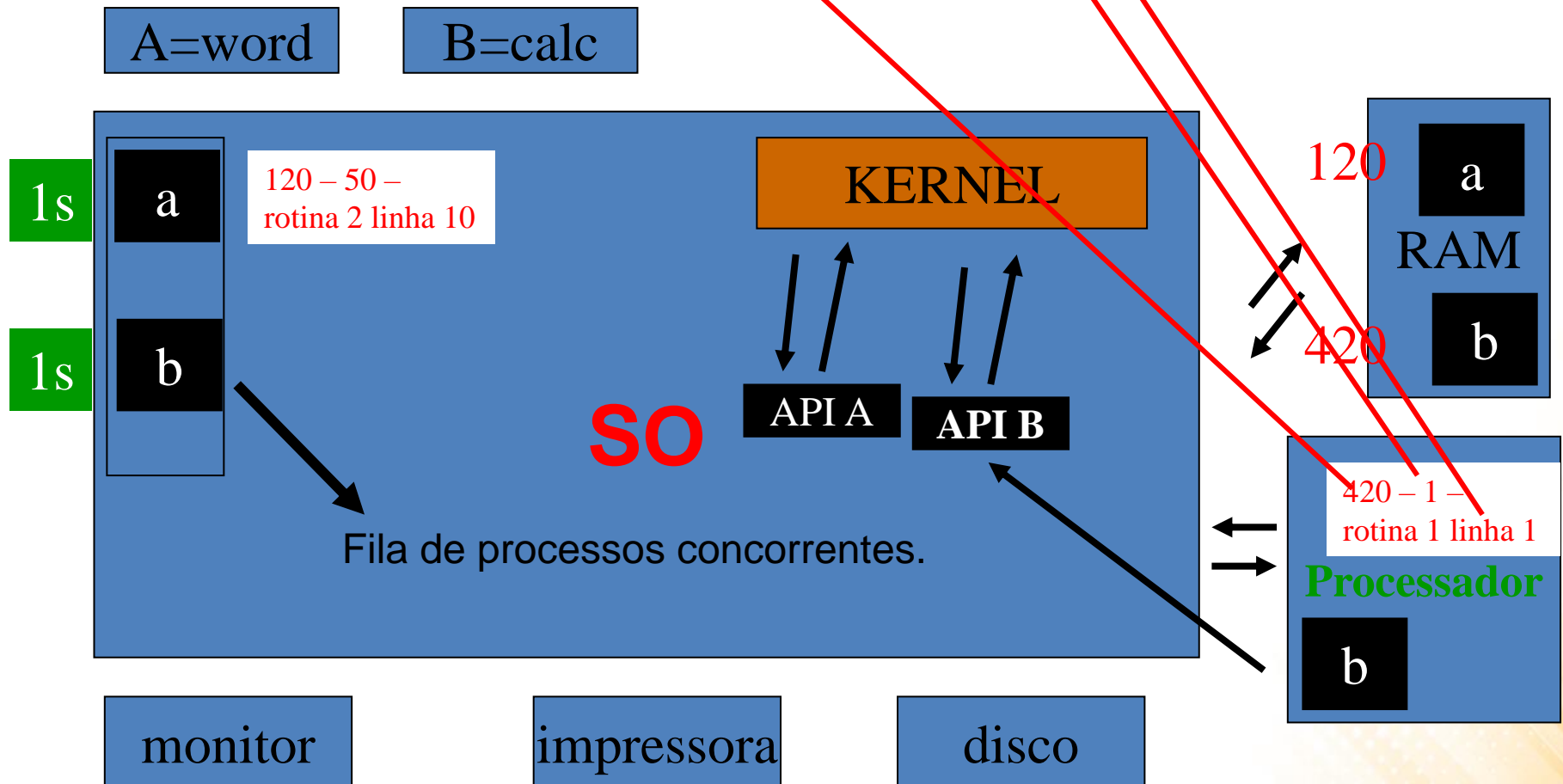
Tempo 2.



Contexto hardware
Ex. posição memória

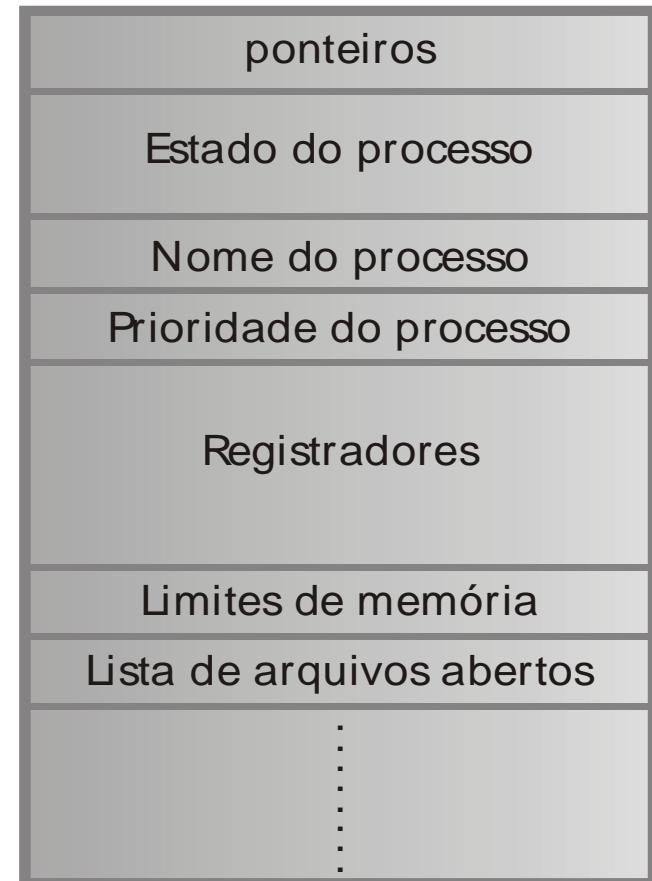
Contexto software
Ex. qtd arquivo simultâneos

Endereçamento
Ex. posição execução (em binário)
Porque não tem código fonte).



Bloco de Controle do Processo

- PCB – informações sobre o processo.



Estados de um processo

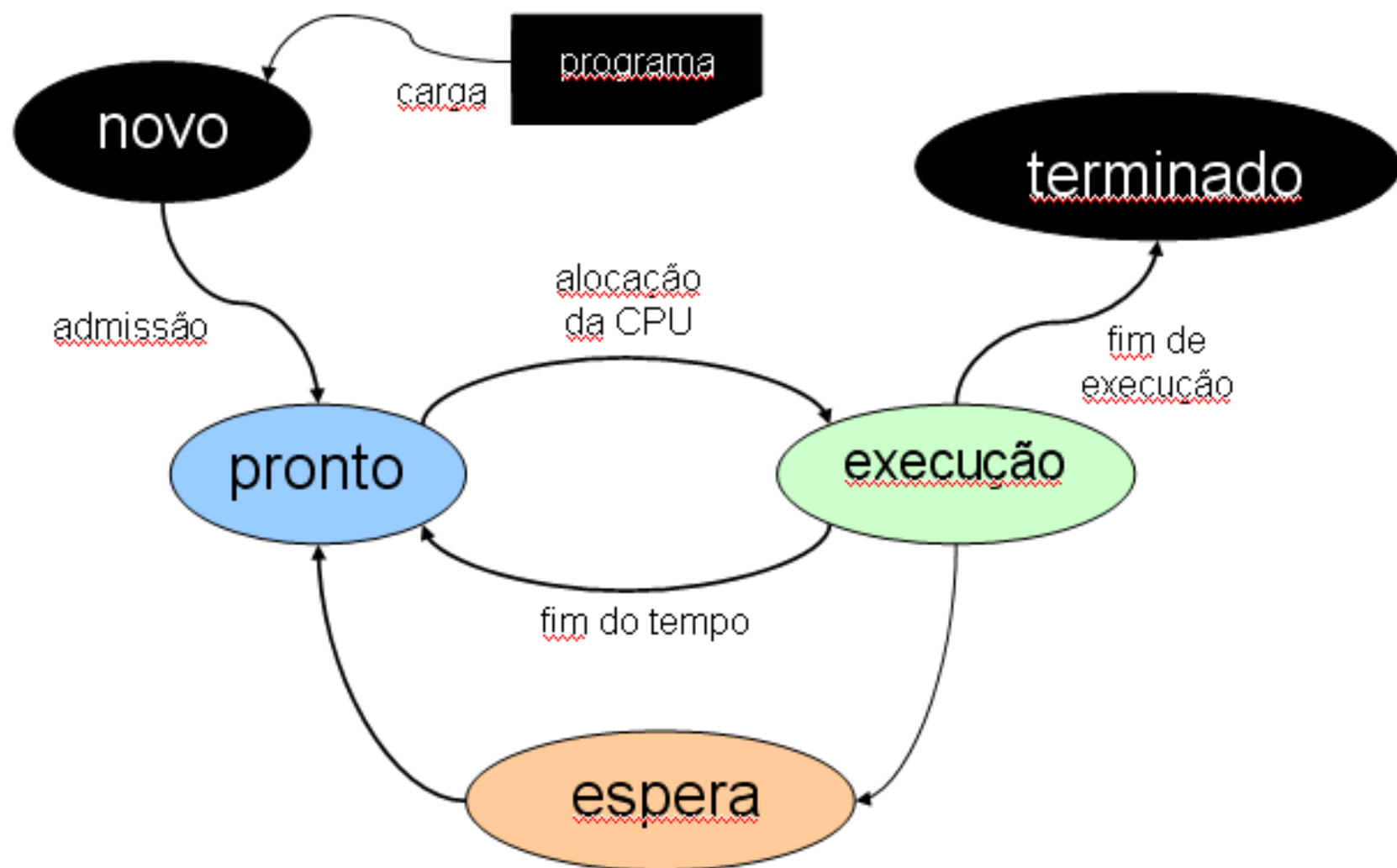
- Cada processo seja uma entidade independente, com seu próprio contador de programa e estado interno, processos muitas vezes precisam interagir entre si. Um processo pode gerar alguma saída que outro processo usa como entrada.

FIGURA 2.2 Um processo pode estar nos estados em execução, bloqueado ou pronto. Transições entre esses estados ocorrem como mostrado.



1. O processo é bloqueado aguardando uma entrada
2. O escalonador seleciona outro processo
3. O escalonador seleciona esse processo
4. A entrada toma-se disponível

1. Em execução (realmente usando a CPU naquele instante).
2. Pronto (executável, temporariamente parado para deixar outro processo ser executado).
3. Bloqueado (incapaz de ser executado até que algum evento externo aconteça).



Mudanças de estado

- **Pronto p/ execução**
 - Ao criar o processo é colocado lista de pronto, onde ele espera por uma oportunidade para ser executado.
 - SO tem próprios algoritmos para escolha da ordem (escalonamento).
- **Execução p/ espera** – o processo passa para espera através de eventos como operação ES.
- **Espera p/ pronto** – Fim de I/O. Um processo no estado de espera sempre terá que passar para o estado de pronto antes de poder ser executado.
- **Execução p/ pronto** – Após fim do tempo de execução ele volta para o estado de pronto, aguardando nova oportunidade.

Implementação de Processos

- Para implementar o modelo de processos, o sistema operacional mantém uma tabela (estrutura de dados) chamada de tabela de processos, com uma entrada para cada um deles. (Alguns autores chamam essas entradas de blocos de controle de processo).
- A Figura no próximo slide mostra alguns dos campos fundamentais em um sistema típico: os campos na primeira coluna relacionam-se ao gerenciamento de processo. Os outros dois relacionam-se ao gerenciamento de memória e de arquivos, respectivamente. Deve-se observar que precisamente quais campos cada tabela de processo tem é algo altamente dependente do sistema, mas esse número dá uma ideia geral dos tipos de informações necessárias.

Implementação de Processos

FIGURA 2.4 Alguns dos campos de uma entrada típica na tabela de processos.

Gerenciamento de processo	Gerenciamento de memória	Gerenciamento de arquivo
Registros Contador de programa Palavra de estado do programa Ponteiro da pilha Estado do processo Prioridade Parâmetros de escalonamento ID do processo Processo pai Grupo de processo Sinais Momento em que um processo foi iniciado Tempo de CPU usado Tempo de CPU do processo filho Tempo do alarme seguinte	Ponteiro para informações sobre o segmento de texto Ponteiro para informações sobre o segmento de dados Ponteiro para informações sobre o segmento de pilha	Diretório-raiz Diretório de trabalho Descritores de arquivo ID do usuário ID do grupo

Threads

- Em sistemas operacionais tradicionais, cada processo tem um espaço de endereçamento e um único thread de controle. Na realidade, essa é quase a definição de um processo. Não obstante isso, em muitas situações, é desejável ter múltiplos threads de controle no mesmo espaço de endereçamento executando em quase paralelo, como se eles fossem (quase) processos separados (exceto pelo espaço de endereçamento compartilhado).

Utilização de threads

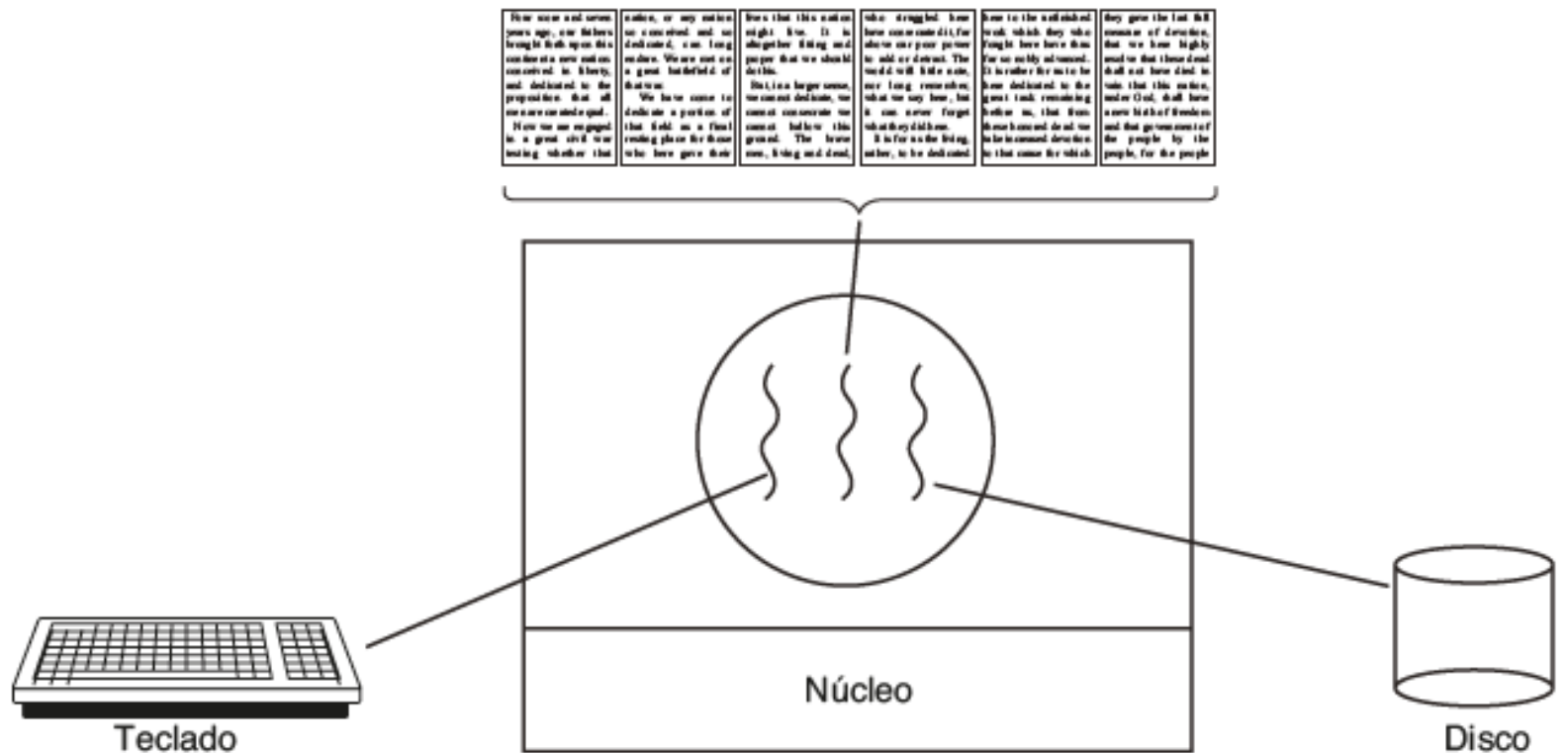
- Por que alguém iria querer ter um tipo de processo dentro de um processo?
 - Na realidade, há várias razões para a existência desses mini processos, chamados threads.
- A principal razão para se ter threads é que em muitas aplicações múltiplas atividades estão ocorrendo simultaneamente e algumas delas podem bloquear de tempos em tempos.
- Ao decompormos uma aplicação dessas em múltiplos threads sequenciais que são executados em quase paralelo, o modelo de programação torna-se mais simples.

Utilização de threads

- Um segundo argumento para a existência dos threads é que como eles são mais leves do que os processos, eles são mais fáceis (isto é, mais rápidos) para criar e destruir do que os processos. Em muitos sistemas, criar um thread é algo de 10 a 100 vezes mais rápido do que criar um processo.
- Uma terceira razão para a existência de threads também é o argumento do desempenho. O uso de threads não resulta em um ganho de desempenho quando todos eles são limitados pela CPU, mas quando há uma computação substancial e também E/S substancial, contar com threads permite que essas atividades se sobreponham, acelerando desse modo a aplicação.

Exemplo de Utilização

Um processador de texto com três threads.



Exemplo de Utilização

- Como um primeiro exemplo, considere um processador de texto. Processadores de texto em geral exibem o documento que está sendo criado em uma tela formatada exatamente como aparecerá na página impressa. Muitos processadores de texto têm a capacidade de salvar automaticamente o arquivo inteiro para o disco em intervalos de poucos minutos para proteger o usuário contra o perigo de perder um dia de trabalho caso o programa ou o sistema trave ou falte energia.
- Deve ficar claro que ter três processos em separado não funcionaria aqui, pois todos os três threads precisam operar no documento. Ao existirem três threads em vez de três processos, eles compartilham de uma memória comum e desse modo têm acesso ao documento que está sendo editado. Com três processos isso seria impossível.

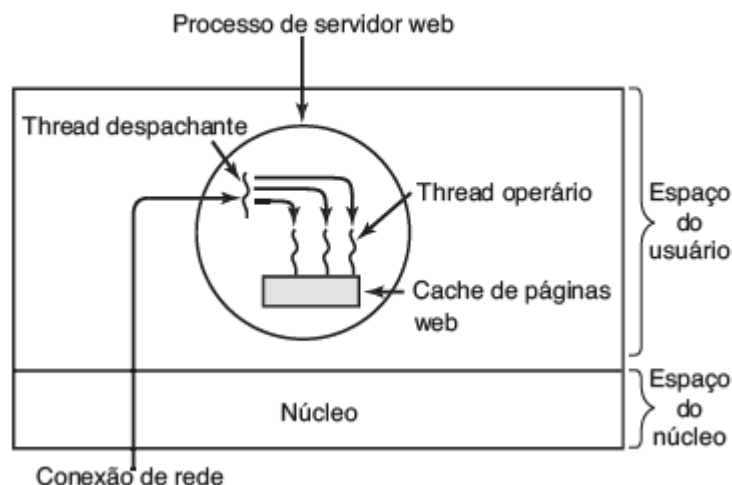
Exemplo de Utilização

- Considere mais um exemplo onde os threads são úteis: um servidor para um website. Solicitações para páginas chegam e a página solicitada é enviada de volta para o cliente. Na maioria dos websites, algumas páginas são mais acessadas do que outras. Por exemplo, a página principal da Sony é acessada muito mais do que uma página mais profunda na árvore contendo as especificações técnicas de alguma câmera em particular. Servidores da web usam esse fato para melhorar o desempenho mantendo uma coleção de páginas intensamente usadas na memória principal para eliminar a necessidade de ir até o disco para buscá-las. Essa coleção é chamada de cache e é usada em muitos outros contextos também.

Exemplo de Utilização

- Uma maneira de organizar o servidor da web é mostrada na figura abaixo. Aqui, um thread, o despachante, lê as requisições de trabalho que chegam da rede. Após examinar a solicitação, ele escolhe um thread operário ocioso (isto é, bloqueado) e passa para ele a solicitação, possivelmente escrevendo um ponteiro para a mensagem em uma palavra especial associada com cada thread. O despachante então acorda o operário adormecido, movendo-o do estado bloqueado para o estado pronto.

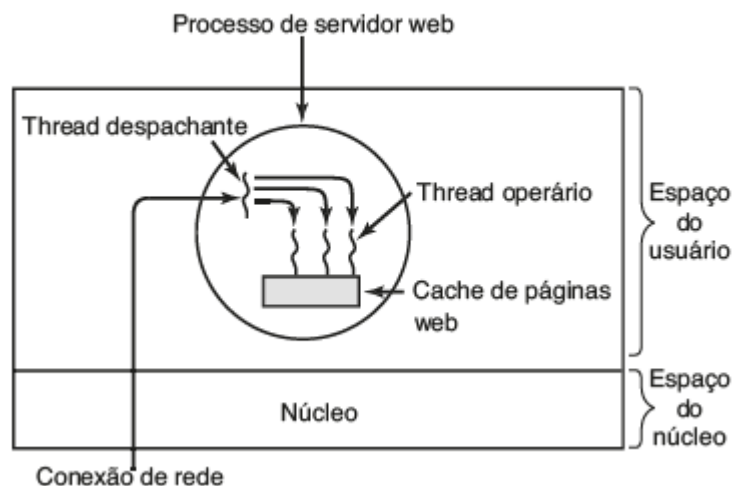
FIGURA 2.8 Um servidor web multithread.



Exemplo de Utilização

- Quando o operário desperta, ele verifica se a solicitação pode ser satisfeita a partir do cache da página da web, ao qual todos os threads têm acesso. Se não puder, ele começa uma operação read para conseguir a página do disco e é bloqueado até a operação de disco ser concluída. Quando o thread é bloqueado na operação de disco, outro thread é escolhido para ser executado, talvez o despachante, a fim de adquirir mais trabalho, ou possivelmente outro operário esteja pronto para ser executado agora. Esse modelo permite que o servidor seja escrito como uma coleção de threads sequenciais.

FIGURA 2.8 Um servidor web multithread.



Exemplo de Utilização

- Um terceiro exemplo em que threads são úteis encontra-se nas aplicações que precisam processar grandes quantidades de dados. Uma abordagem normal é ler em um bloco de dados, processá-lo e então escrevê-lo de novo. O problema aqui é que se houver apenas a disponibilidade de chamadas de sistema bloqueantes, o processo é bloqueado enquanto os dados estão chegando e saindo. Ter uma CPU ociosa quando há muita computação a ser feita é um claro desperdício e deve ser evitado se possível.
- Threads oferecem uma solução: o processo poderia ser estruturado com um thread de entrada, um de processamento e um de saída. O thread de entrada lê dados para um buffer de entrada; o thread de processamento pega os dados do buffer de entrada, processa-os e coloca os resultados no buffer de saída; e o thread de saída escreve esses resultados de volta para o disco. Dessa maneira, entrada, saída e processamento podem estar todos acontecendo ao mesmo tempo.

Exercícios

1. Explique o conceito de processos.
2. Quais os estados de um processo?
3. Com a relação a mudança de estado, explique:
 - a) Quando um processo passa do estado de pronto para execução.
 - b) Quando um processo passa do estado de execução para espera.
 - c) Quando um processo passa do estado de espera para pronto.
 - d) Quando um processo passa do estado de execução para pronto.
4. Cite e explique as três partes que formam um processo.
5. Explique o que são threads e dê um exemplo de utilização.

Referência desta Aula

- TANENBAUM, A. S. Sistemas Operacionais Modernos, Prentice-Hall do Brasil, 4ª edição, 2016.

Fim
Obrigado