Chp. 3 - Process Description and Control

... Chp. 3 - Process Description and Control

✯ William STALLINGS; **Operating Systems: Internals and Design Principles**, New Jerssey, Prentice-Hall, 1998, ISBN: 0-13-887407-7

✯ Eleri CARDOZO; Maurício MAGALHÃES; Luís F. FAINA; **Sistemas Operacionais**, Dep. de Eng. de Computação e Automação Industrial, Fac. de Engenharia Elétrica e de Computação, UNICAMP, 1996.

Chp. 3 - Descrição e Controle de Processos

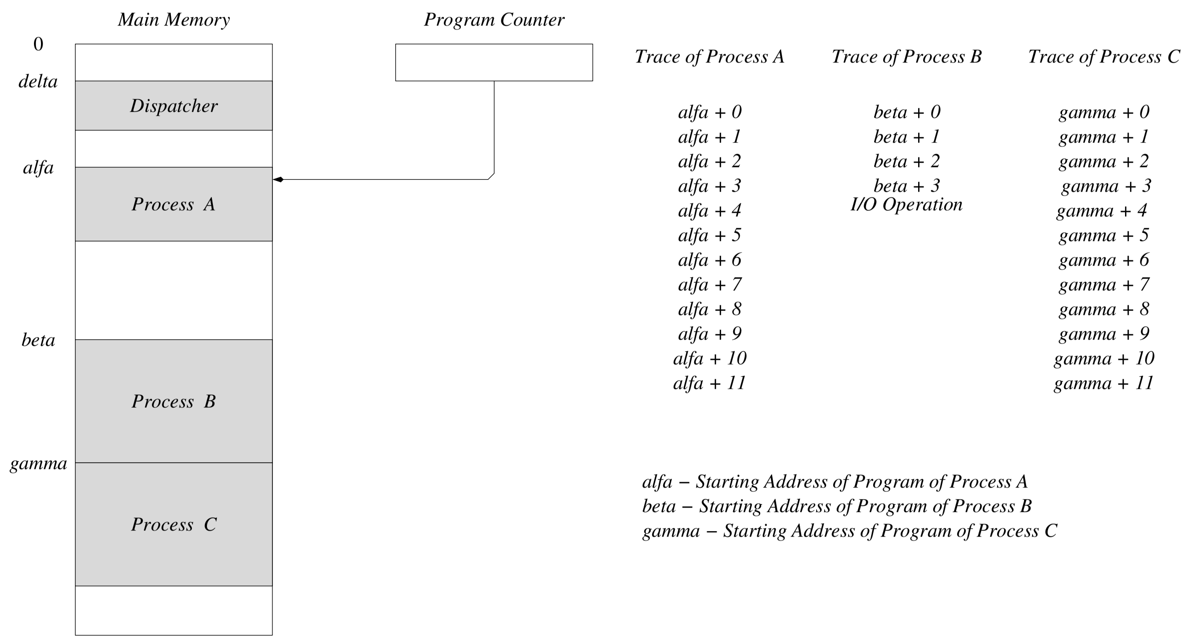
✰ **projeto de um sistema operacional:** deve refletir os requisitos para os quais ele foi projetado, por exemplo, um sistema operacional que suporta multiprogramação deve suportar algumas centenas de usuários;

- ... muitos dos requisitos que os sistemas operacionais precisam suportar podem ser expressos e/ou desenvolvidos em torno do conceito de processos:
  - ✯ o sistema operacional deve entrelaçar a execução de vários processos;
  - ✯ o sist. oper. deve alocar recursos aos processos segundo políticas estabelecidas;
  - ✯ o sistema operacional pode ser requisitado pelo usuário a criar processos ou suportar a comunicação entre processos.

- ... iremos começar o estudo detalhado de sistemas operacionais examinando a maneira pela qual eles representam e controlam os processos.

3.1 - Estados de um Processo

✰ **principal função de um processador:** é executar instruções de máquina residentes na memória principal — instruções são providas na forma de um programa.



... 3.1 - Estados de um Processo

- ... vejamos a execução dos mesmos passos anteriormente apresentados do ponto de vista do processador, considerando que o sistema operacional permite a um processo continuar sua execução por no máximo 06 ciclos de instruções.

1	*alfa + 0*	
2	*alfa + 1*	
3	*alfa + 2*	
4	*alfa + 3*	
5	*alfa + 4*	
6	*alfa + 5*	
	----------	*Time Out*
7	*delta + 0*	
8	*delta + 1*	
9	*delta +2*	
10	*delta + 3*	
11	*delta + 4*	
12	*delta + 5*	
13	*beta + 0*	
14	*beta + 1*	
15	*beta + 2*	
16	*beta + 3*	
	----------	*I/O Operation*
17	*delta + 0*	
18	*delta + 1*	
19	*delta +2*	
20	*delta + 3*	

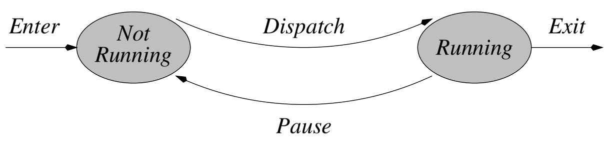
21	*delta + 4*	
22	*delta + 5*	
23	*gamma + 0*	
24	*gamma + 1*	
25	*gamma + 2*	
26	*gamma + 3*	
27	*gamma + 4*	
28	*gamma + 5*	
	----------	*Time Out*
29	*delta + 0*	
30	*delta + 1*	
31	*delta +2*	
32	*delta + 3*	
33	*delta + 4*	
34	*delta + 5*	
35	*alfa + 6*	
36	*alfa + 7*	
37	*alfa + 8*	
38	*alfa + 9*	
39	*alfa + 10*	
40	*alfa + 11*	
	----------	*Time Out*

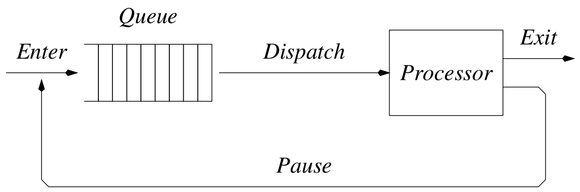
41	*delta + 0*	
42	*delta + 1*	
43	*delta +2*	
44	*delta + 3*	
45	*delta + 4*	
46	*delta + 5*	
47	*gamma + 6*	
48	*gamma + 7*	
49	*gamma + 8*	
50	*gamma + 9*	
51	*gamma + 10*	
52	*gamma + 11*	
	----------	*Time Out*

delta - Starting Address of Program of Process

... 3.1 - Estados de um Processo - Modelo de 02 Estados

- ... a principal responsabilidade do sistema operacional é o controle da execução dos processos, que por sua vez inclui determinar o padrão com o qual os processos são entrelaçados bem como a alocação de recursos requisitados.





- ... o modelo mais simples é aquele que contempla o processo em pelo menos dois estados: processo está sendo executado ou não pelo processador;

- ... assim o processo pode estar em um dos 02 estados: *running* ou *not running*;

- ... processos que não estão sendo executados devem ser mantidos em uma fila, esperando sua vez para serem executados.

... 3.1 - Estados de um Processo - Criação e Finalização

- **criação de um processo** - quando um novo processo é criado, o sistema operacional reserva espaço em memória para estruturas de dados que serão usadas para gerenciar o processo, além de alocar espaço em memória para o processo.

- ... 04 eventos comuns conduzem a criação de um processo, como indicado abaixo:

  ❶ em um sistema *batch*, um processo é criado em resposta à submissão de tarefas;
  ❷ em um ambiente interativo, um processo é criado quando um novo usuário executa o *log in*, ou seja, o usuário abre um sessão de trabalho;
  ❸ um sistema operacional pode criar um processo para executar funções em favor de um programa de usuário, sem que o usuário tenha que esperar (p. ex., impressão);
  ❹ com o propósito de oferecer modularidade ou explorar o paralelismo, um programa de usuário pode inquirir a criação de processos.

- ... quando um processo cria outro, o primeiro é referenciado como um processo pai e o processo criado como processo filho — tipicamente processos com parentesco necessitam comunicar e cooperar uns com os outros.

... 3.1 - Estados de um Processo - Criação e Finalização

- ... razões possíveis que conduzem a finalização de um processo:

  ✣ conclusão normal - o processo executa do sistema operacional uma chamada de serviço para informar que ele concluiu sua execução;
  ✣ tempo limite excedido - o processo utilizou o processador por mais tempo que o total especificado, p. ex., *wall clock time*, tempo total executando e etc.;
  ✣ memória indisponível - o processo requer mais memória que o sist. pode prover;
  ✣ violação de limites - o processo tenta acessar um posição de memória cujo acesso não está habilitado para aquele processo;
  ✣ erro de proteção - um processo tenta utilizar um recurso ou um arquivo que não lhe é permitido, p. ex., escrever em um arquivo com permissão apenas para leitura;
  ✣ erro aritmético - o processo solicita um operação proibida, p. ex., divisão por zero, armazenamento de número maior que o permitido;
  ✣ tempo excedido - o processo aguarda um tempo superior ao máximo especificado para que um determinado evento ocorra;

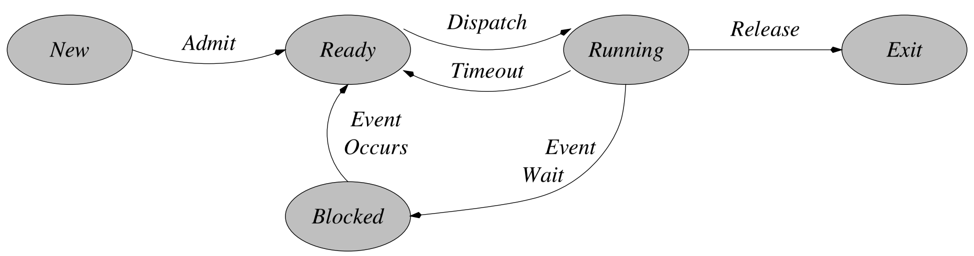
... 3.1 - Estados de um Processo - Criação e Finalização

- ... razões possíveis que conduzem a finalização de um processo:
  - falha de entrada/saída - um erro ocorre durante um operação de entrada/saída, p. ex., inabilidade de achar um arquivo;
  - instrução inválida - o processo solicita a execução de uma instrução que não existe, p. ex., um salto para uma área de dados e posterior execução;
  - instrução com privilégio - o processo solicita a execução de uma instrução reservada para o sistema operacional;
  - mal uso de dados - um dado não foi inicializado ou é de um tipo não definido;
  - intervenção do sist. oper. ou operador - p. ex., em um *deadlock*;
  - término do pai - quando o processo pai termina, o sistema operacional pode automaticamente terminar todos os seus descendentes;
  - requisição do pai - o processo pai tipicamente tem a autoridade para terminar qualquer um de seus descendentes.

... 3.1 - Estados de um Processo - Modelo de 05 Estados

- como apresentado no modelo de 02 estados, a fila de processos que não estão executando contém tanto processos pronto para serem executados quanto processos bloqueados, o que sugere 02 novos estados: *ready* e *blocked*;

- ... 02 outros estados, *new* e *exit* são adicionados, pois constituem blocos úteis para o gerenciamento de processos:
  - o estado *new* corresponde a um processo que acabou de ser definido;
  - o estado *exit* corresponde ao estado intermediário pelo qual o processo passa antes que a estrutura de dados usada para o seu gerenciamento seja liberada.



... 3.1 - Estados de um Processo - Modelo de 05 Estados

- ... para o exemplo anteriormente apresentado, Processos A, B e C, a tabela abaixo apresenta as transições de cada processo pelos 05 estados:

Time	Processo A	Processo B	Processo C
1-6	***Running***	*Ready*	*Ready*
7-12	*Ready*	*Ready*	*Ready*
13-16	*Ready*	***Running***	*Ready*
17-22	*Ready*	*Blocked*	*Ready*
23-28	*Ready*	*Blocked*	***Running***
29-34	*Ready*	*Blocked*	*Ready*
35-40	***Running***	*Blocked*	*Ready*
41-46	*Ready*	*Blocked*	*Ready*
47-52	*Ready*	*Blocked*	***Running***

... 3.1 - Estados de um Processo - Processo Suspenso

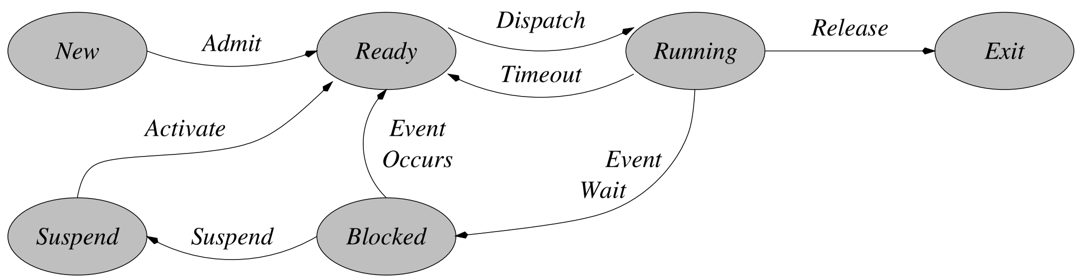
- ... os 03 estados anteriormente descritos (*ready*, *running* e *blocked*) provêem um modo sistemático para modelar o comportamento dos processos, bem como guiar sua implementação em um sistema operacional;

- ... no entanto, há justificativas para acrescentarmos mais estados:
  - cada processo para ser executado deve estar completamente carregado na memória principal e todos os processos na fila devem estar residentes na memória;
  - como acomodar mais processos na memória quando esta está no seu limite ?

- ... algumas possíveis soluções para este problema:
  - expandir a memória principal para acomodar mais processos;
  - mover partes ou todo o processo da memória principal para um memória secundária, ou seja, realizar a operação de *swapping*.

... 3.1 - Estados de um Processo - Processo Suspenso

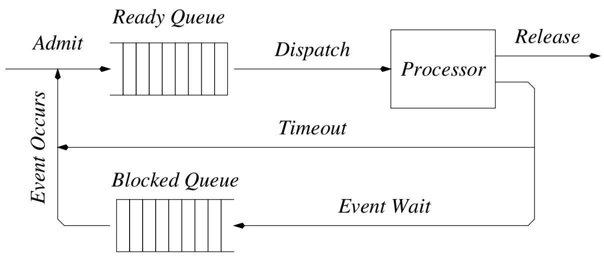
- ... se utilizarmos *swapping* como descrito anteriormente, um novo estado deve ser adicionado ao nosso modelo que descreve o comportamento de um processo:

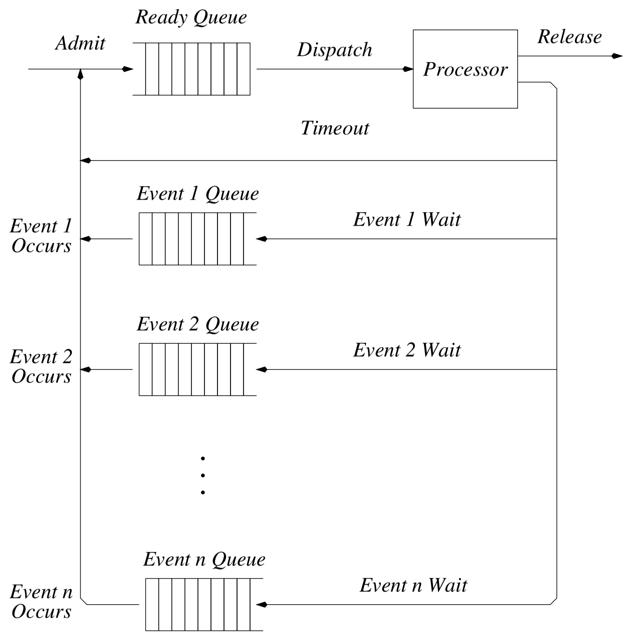


- ... este modelo, no entanto, apresenta alguns problemas:
  - ✘ há duas escolhas para se trazer um processo para o estado de pronto: os sistema pode admitir um processo no estado *new* ou buscar um no estado *suspend*;
  - ✘ quando o sistema operacional trazer um processo suspenso para a memória principal, ele estará trazendo um processo bloqueado — como justificar ?

... 3.1 - Estados de um Processo - Processo Suspenso

- ... nas múltiplas filas de processos bloqueados, todos eles em todas as filas precisam estar residentes na memória, assim, como acomodar um novo processo ?

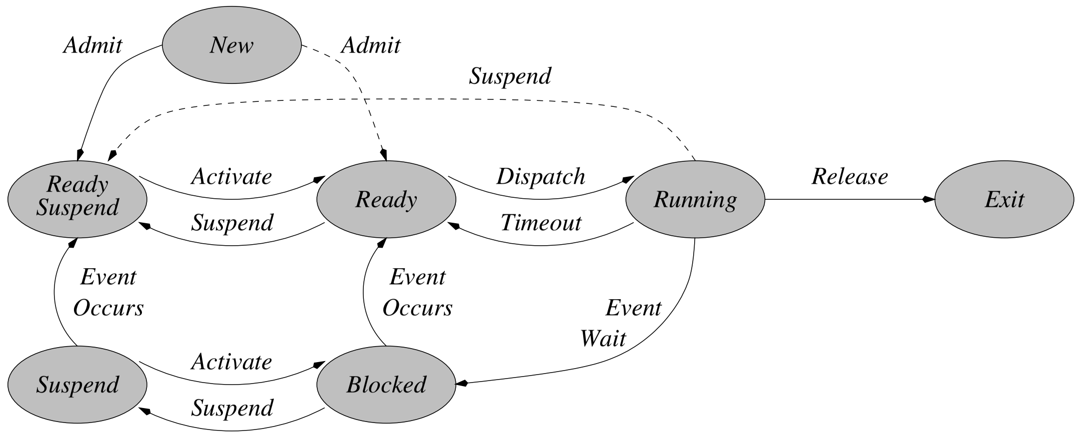




... 3.1 - Estados de um Processo - Processo Suspenso

- quando o sistema operacional trazer um processo suspenso para a memória principal, ele estará trazendo um processo bloqueado — como justificar ?

- ... desde modo temos que repensar este aspecto de projeto, pois temos 02 conceitos diferentes: suspenso ou não suspenso e bloqueado ou não bloqueado.



... 3.1 - Estados de um Processo - Processo Suspenso

- ... nós podemos generalizar o conceito de processo suspenso, definindo tais processos como aqueles como as seguintes características:

  ❶ o processo não está imediatamente disponível para execução;
  ❷ o processo pode ou não estar esperando por um evento; e caso esteja, sua condição de bloqueado independe da condição de suspenso e em sendo assim, a ocorrência de um evento não habilita o processo a ser executado;
  ❸ o processo foi suspenso por um agente: por ele mesmo, pelo processo pai ou pelo sistema operacional com o propósito de prevení-lo de ser executado;
  ❹ o processo pode não ser removido de seu estado até que o agente ordene explicitamente a remoção desse estado.

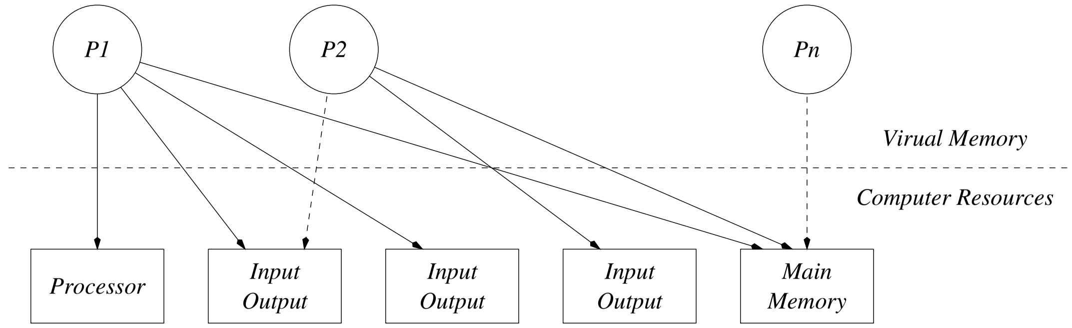
- ... algumas razões para suspender um processo:

  ❖ *swapping* - o sist. oper. necessitar liberar espaço na memória principal;
  ❖ temporização - o processo pode ser executado periodicamente e pode ser suspenso enquanto esperando pelo próximo intervalo de execução.

3.2 - Descrição de Processos

- ... fundamentalmente nós podemos conceber o sistema operacional como uma entidade que gerencia o uso de recursos do sistema através dos processos;

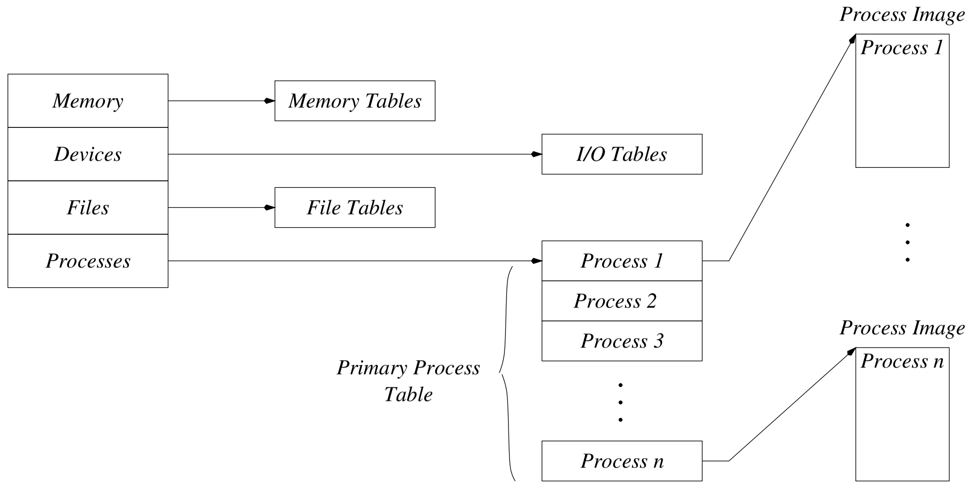
- ... este conceito é ilustrado na figura abaixo, que descreve um ambiente multiprogramado com um número de Processos (P1, P2, ... Pn) que foram criados e que ocupam parte da memória principal.



- ... processos P1 e P2 estão na memória principal, embora P2 esteja bloqueado esperando por um dispositivo de entrada/saída alocado por P1. P3 está suspenso.

... 3.2 - Descrição de Processos - Estrutura de Controle

- ... para o sistema operacional gerenciar processos e recursos, ele deve ter informações sobre o *status* corrente de cada processo e recurso — para isto, o sist. oper. instância e mantém tabelas de informação sobre estas entidades.
- ... fundamentalmente, um sistema operacional mantém informações em quatro categorias: memória, entrada/saída, arquivo e processo.



... 3.2 - Descrição de Processos - Estrutura de Controle

- ... para gerenciar e controlar um processo, o sistema operacional precisa conhecer: onde está localizado o processo e os seus atributos (*process control block*).
- um processo deve no mínimo incluir um programa(s) para ser executado, e pode estar associado a um conjunto de dados, englobar uma pilha e ter atributos;
- ... esta coleção é comumente referenciada como imagem de um processo.
- Em relação aos atributos, podemos agrupar as informações do bloco de controle de processos em três categorias gerais: identificação do processo, informação do estado do processador e informação de controle do processo.
- **Identificação do Processo** - praticamente em todos os sistemas operacionais um número de identificação única é atribuído a cada processo.
  - ... dentre os identificadores estão: identificador do processo, identificador do processo que o criou (processo pai) e identificador do usuário.

... 3.2 - Descrição de Processos - Estrutura de Controle

- **Informação do Estado do Processador** - consiste dos conteúdos dos registradores visíveis para o usuário, registradores de controle (p. ex., contador de programa), registrador *program status word* e apontadores de pilha.
- **Informações de Controle do Processo** - esta é a informação adicional necessária para que o sistema operacional controle e coordene os vários processos;
- ... neste conjunto destacamos:
  - ✯ informações de estado e escalonamento - p. ex., estado do processo, prioridade, informações relativas ao escalonamento e eventos;
  - ✯ estrutura de dados - p. ex, fila de processos;
  - ✯ comunicação entre processos - vários *flags*, sinais e mensagens podem estar associadas à comunicação entre processos;
  - ✯ privilégios do processo - processos podem ter privilégios em relação a memória acessada e aos tipos de instruções que podem ser executados;

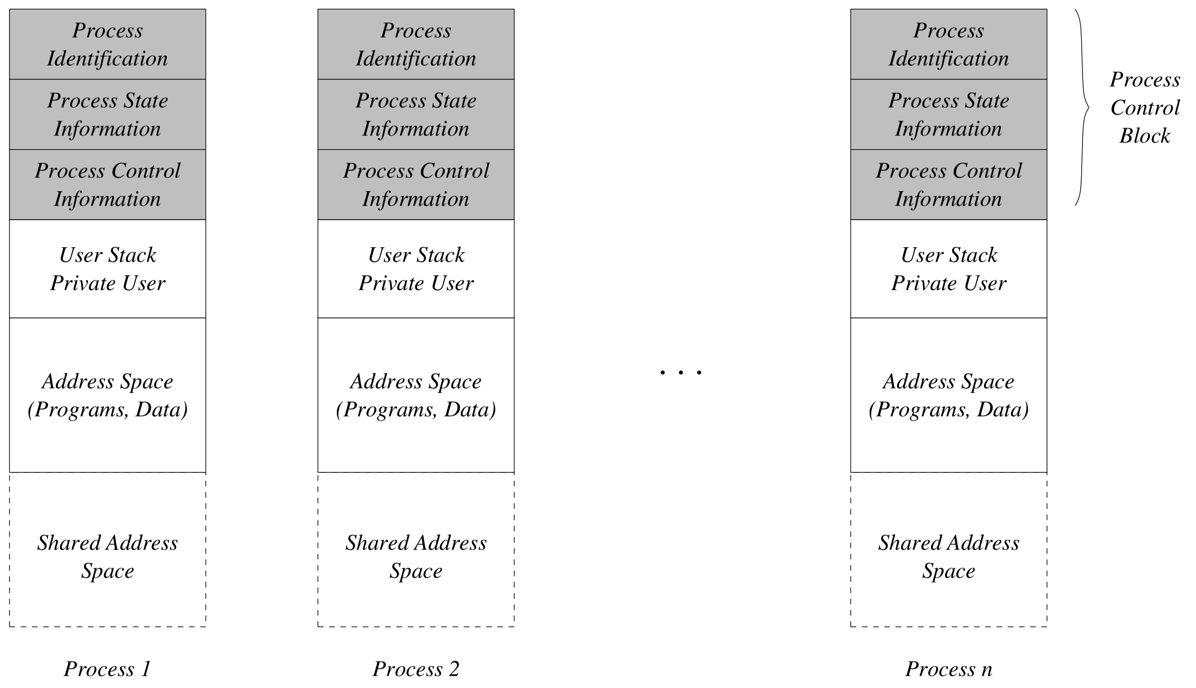
... 3.2 - Descrição de Processos - Estrutura de Controle

- ... neste conjunto destacamos:
  - ✭ gerenciamento de memória - esta sessão inclui ponteiros tabelas de segmentos e/ou páginas que descrevem a memória virtual atribuída ao processo;
  - ✭ utilização e propriedade de recursos - recursos controlados por um processo podem ser indicados, p. ex., arquivos abertos bem como a história de utilização do processador e de outros recursos.

- Portanto, o bloco de controle de um processo contém todas as informações necessárias sobre o processo para que o sistema operacional possa gerenciá-lo e controlá-lo.

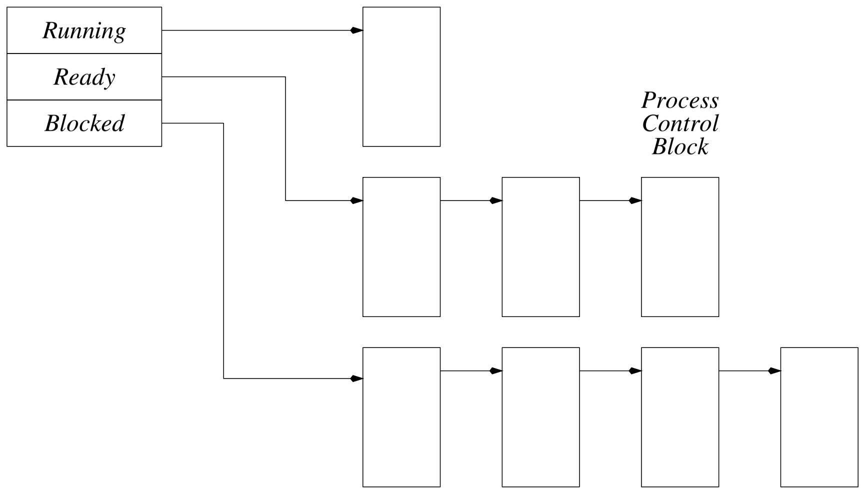
... 3.2 - Descrição de Processos - Estrutura de Controle

- ... a figura abaixo sugere a estrutura da imagem de um processo na memória virtual:



... 3.2 - Descrição de Processos - Estrutura de Controle

- ... para ilustrar a estrutura de uma lista de processos, considere alguns processos cujos estados estão distribuídos entre *ready*, *blocked* e *running*.



- **Aspecto de Projeto** - rotinas dentro do sistema operacional se farão necessárias para acessar as informações dos blocos de controle dos processos !

3.3 - Controle de Processos - Modos de Execução

- ... a maioria dos processos suportam pelo menos 02 modos de execução: modo usuário (modo típico de execução) e modo núcleo (ou com privilégios).
- tipicamente, as funções presentes no *Kernel* do Sistema Operacional, são: gerenciamento de processos, memória, entrada/saída e funções de suporte.

★ Gerenciamento de Processos - criação e finalização; escalonamento e expedição; chaveamento entre processos; sincronização e suporte a comunicação entre processos e gerenciamento dos blocos de controle dos processos;

★ Gerenciamento de Memória - alocação de espaço de endereçamento para processos; *swapping* e gerenciamento de páginas e segmentos;

★ Gerenciamento de Entrada/Saída - gerenciamento de *buffer* e alocação de canais e dispositivos de entrada/saída para processos;

★ Funções de Suporte - manipulação de interrupções; contabilidade e monitoramento.

... 3.3 - Controle de Processos - Modos de Execução

- ... a razão para se utilizar 02 modos é a necessidade de proteger o sistema operacional e suas tabelas chaves, tais quais os blocos de controle de processos.

- ... no modo *kernel*, o *software* tem total controle do processador e de todas as suas instruções, registros e memória — assim este nível de controle não é necessário, por segurança, e não desejável para programas de usuário.

- ?! como o processador sabe em que modo ele está executando e como mudá-lo ?!
  - ✯ a primeira questão é respondida com auxílio do registro de *program status word*, que contém um bit indicando o modo;
  - ✯ para mudar o modo, a mudança ocorre em resposta a certos eventos, p. ex., quando o usuário invoca um serviço do sistema operacional.

... 3.3 - Controle de Processos - Criação de Processos

- Uma vez que o Sistema Operacional decida por criar um processo, seja por que razão for, isto pode ser feito da seguinte maneira:
  - ❶ atribuir um único identificador ao novo processo;
  - ❷ alocar espaço para o processo (elementos da imagem do processo) - o sist. oper. deve saber quanto espaço para o endereçamento (código e dados) bem como para a pilha (p. ex., estes valores podem ser previamente definidos);
  - ❸ inicialização do bloco de controle do processo - isto inclui os campos de identificação do processo, informações do estado do processador e informações de controle do processo;
  - ❹ tendo a estrutura de dados sido inicializada, o sist. oper. precisa estabelecer as referências apropriadas entre os vários blocos e tabelas de informações;
  - ❺ criação e expansão de outras estruturas de dados a serem utilizadas, por exemplo, na manutenção de um arquivo de contabilidade para cada processo com o propósito de precificação bem como avaliar a sua performance.

... 3.3 - Controle de Processos - Chaveamento de Processos

- idéia do chaveamento de processos - em algum instante, um processo executando é interrompido e o sistema operacional atribui outro processo ao estado de executando e muda o controle para aquele processo.

- ... mas como o sist. oper. readquire o controle para efetuar o chaveamento ?!

- Primeiro, o chaveamento de processos pode ocorrer a qualquer instante que o sist. oper. tenha readquirido o controle do processo que está executando correntemente:

  ✧ interrupção - é um sinal externo a execução da instrução corrente;
  ✧ *trap* - associado com a execução da instrução corrente;
  ✧ requisição explícita - chamada de função do sist. operacional.

- ... com uma interrupção, o controle é inicialmente transferido para um manipulador de interrupção, que por sua vez executa algum procedimento básico e depois transfere para a rotina do sist. oper. responsável pela tratamento da interrupção em particular.

... 3.3 - Controle de Processos - Chaveamento de Processos

- ... outro aspecto a ser investigado é como reconhecer e distinguir entre modo de chaveamento e chaveamento entre processos ?!

- Inicialmente, deve esta claro que o modo de chaveamento (modo usuário ou modo *kernel*) é um modo distinto do chaveamento de processos.

- Segundo, o modo de chaveamento pode ocorrer sem que haja mudança do estado do processo que está sendo executado — neste caso, o *overhead* é baixo.

- ... entretanto, se o processo correntemente executando tem seu estado alterado (*ready*, *blocked*, *ready* e suspenso, etc.), então o sistema operacional deverá efetuar mudanças substanciais no seu ambiente.

... 3.3 - Controle de Processos - Chaveamento de Processos

- os passos envolvidos em um chaveamento completo de processos são os seguintes:

❶ salvar o contexto do processo, incluindo contador de programa e registradores;

❷ atualizar o bloco de controle do processo para o processo que está sendo executado, por exemplo, mudar o estado do processo;

❸ mover o bloco de controle do processo para a fila correspondente;

❹ selecionar outro processo para execução segundo algum critério;

❺ atualizar o bloco de controle do processo do processo selecionado;

❻ atualizar a estrutura de dados do gerenciamento de memória;

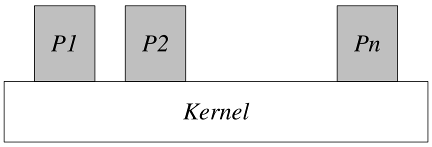
❼ recuperar o contexto do processador no que se refere ao processo que acaba de ser selecionado, ou seja, carregar os valores prévios para o contador de programa e demais registradores do processo selecionado.

... 3.3 - Controle de Processos - Execução do S.O.

- ... alguns aspectos anteriormente apontados no que se refere a definição do sistema operacional como um programa ordinário, podem agora ser melhor esclarecidos !

- Se isto é verdade, é o Sistema Operacional um processo ? Se for, como ele é controlado ? — estas questões tem recebido muitas respostas por parte dos projetistas e a seguir apresentamos algumas abordagens.

✭ *kernel* separado - quando um processo corrente é interrompido ou requerê uma chamada de sistema, o contexto de modo deste processo é salvo e o controle é passado para o *kernel*, que possui sua particular região de memória e de pilha.



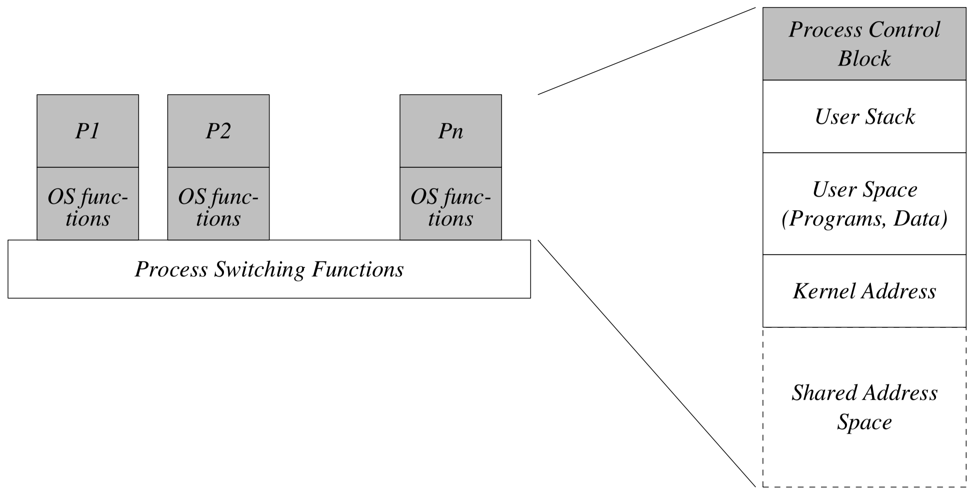
... 3.3 - Controle de Processos - Execução do S.O.

- ... o sistema operacional pode executar qualquer função que deseje e depois recuperar o contexto do processo interrompido que a tenha solicitado;
- uma outra alternativa após o sistema operacional armazenar o ambiente do processo é proceder o escalonamento e despacho de um outro processo;
- ... o direcionamento de escalonar um outro processo ou proceder a execução de uma função depende da razão da interrupção e de circunstâncias temporais;
- ... seja qual for o caso, o ponto chave nesta abordagem é que o conceito de processo é aplicável somente a processos do usuário, ou seja, o código do sistema operacional é executado como uma entidade separada que opera em modo privilegiado.

... 3.3 - Controle de Processos - Execução do S.O.

✯ funções do sist. oper. executam dentro do processo do usuário - alternativa comum dentro de sistemas operacionais característicos dos mini e microcomputadores.

- ... o sist. oper. é uma coleção de rotinas que o usuário chama para realizar várias funções, sendo que são executadas dentro do ambiente do processo do usuário;



... 3.3 - Controle de Processos - Execução do S.O.

- ... como apresentado, uma estrutura típica para imagem de um processo mostra que uma pilha de *kernel* separada é usada para gerenciar as chamadas/retornos enquanto o processo está no modo *kernel*;

- ... códigos e dados do sistema operacional estão em um espaço de endereçamento comum e são compartilhados por todos os processos do usuário;

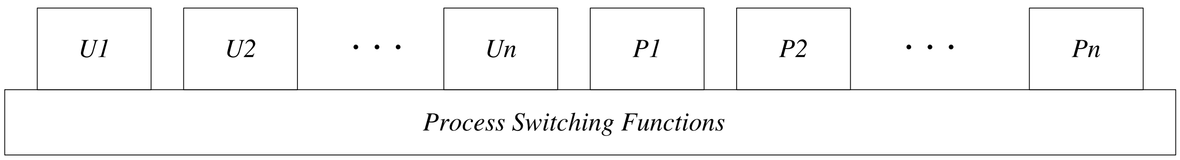
- ... quando uma interrupção, *trap* ou chamada de sistema ocorre, o processador é posto em modo *kernel* e o controle é transferido para o sistema operacional;

- ... isto, no entanto, não significa chaveamento de processo, mas apenas o chaveamento de modo dentro do mesmo processo, ou seja, processo do usuário;

- se, por outro lado, for determinado um chaveamento de processo, então o controle é transferido para a rotina de chaveamento de processos, que pode ou não executar dentro do processo corrente (depende do projeto de sistema);

... 3.3 - Controle de Processos - Execução do S.O.

- uma outra alternativa, como ilustrado abaixo, é a visão do sistema operacional como uma coleção de processos do sistema, ou seja, as funções principais do *kernel* estão organizadas como processos separados;



✔ impõe um projeto de programas disciplinado que encoraja o uso de um sistema operacional modular com o mínimo de interfaces claras entre os módulos;

✔ algumas funções de sistema operacional não críticas são convenientemente implementadas como processos separados permitindo a concorrência;

✔ implementar o sistema operacional como um conjunto de processos é útil em ambientes multiprocessados ou em múltiplos computadores, onde alguns serviços do sistema podem ser enviados para processadores dedicados, melhorando a performance.

3.4 - Gerenciamento de Processos no UNIX SVR 4

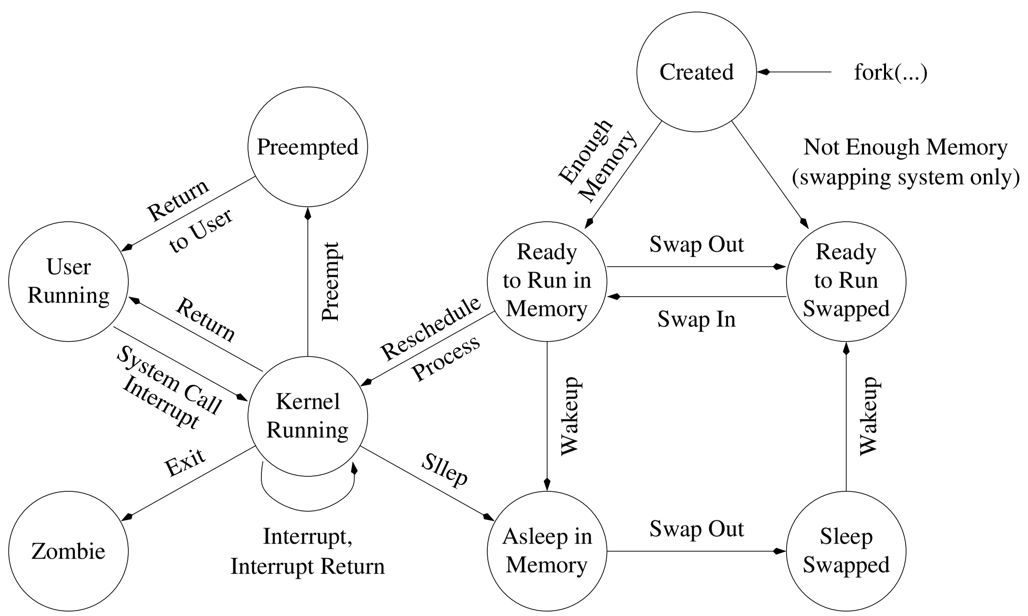
✰ UNIX SVR4 segue a abordagem na qual a maior parte do sistema operacional executa dentro do ambiente dos processos do usuário (modo *kernel* e usuário são necessários);

- ... processos do sistema executam no modo *kernel* e desempenham tarefas administrativas bem como funções de manutenção do sistema, p.ex.: alocação de memória e permuta de processos;

- ... processos do usuário operam no modo usuário para executarem programas do usuário e no modo *kernel* para executar instruções pertencentes ao *kernel*.

... 3.4 - Gerenciamento de Processos no UNIX SVR 4

✰ UNIX reconhece 09 estados de processos abaixo apresentados:



... 3.4 - Gerenciamento de Processos no UNIX SVR 4

✰ UNIX emprega 02 estados de execução para indicar se o processo está executando no modo usuário ou no modo *kernel* (*kernel and user running*).

✰ Distinção entre 02 estados: **Ready to Run in Memory** e **Preempted**. ... essencialmente constituem o mesmo estado, entretanto, isto é feito para enfatizar o modo pelo qual o estado de preempção é alcançado;

- ... quando um processo está executando no modo *kernel*, chegará o instante em que o *kernel* terá completado sua tarefa e retornará o controle para o programa usuário;
- ... neste ponto, o *kernel* pode decidir preemptar o corrente processo em favor de um outro com mais alta prioridade e pronto para executar;
- ... neste caso, o corrente processo move de estado (Preempted).

... 3.4 - Gerenciamento de Processos no UNIX SVR 4

✰ Processo no UNIX é um conjunto complexo de estruturas de dados que possibilita ao sistema operacional informações necessárias para gerenciar e manter processos;

- ... os principais elementos da imagem do processo são organizados em 03 partes: *user-level context*; *register context* e *system-level context*;

	User-Level Context
Process Text	Executable machine instructions of the program.
Process Data	Data accessible by the program of this process.
User Stack	Contains the arguments, local variables, and pointers for functions executing in user mode.
Shared Memory	Memory shared with other processes, used for interprocess communcication

... 3.4 - Gerenciamento de Processos no UNIX SVR 4

	Register Context
Program Counter	Address of next instruction to be executed; may be in kernel or user memory space of this process.
Processor Status Register	Contains the hardware status at the time of preemption; contents and format are hardware dependent.
Stack Pointer	Points to the top of the kernel or user stack, depending on the mode of operation at the time or preemption.
General-purpose Registers	Hardware dependent.

	System-Level Context
Process Table Entry	Defines state of a process; this information is always accessible to the operating system.
U (User) Area	Process control information that needs to be accessed only in the context of the process.
Per Process Region Table	Defines the mapping from virtual to physical address; also contains a permission field that indicates the type of access allowed the process: read-only, read-write, or real-execute.
hline Kernel Stack	Contains the stack frame or kernel procedures as the process executes ini kernel mode.

... 3.4 - Gerenciamento de Processos no UNIX SVR 4

✰ UNIX Process Table Entry:

Process Status	Current state of process.
Ponters	To U area and process memory area (text, data, stack).
Process Size	Enables the operating system to know how much space to allocate the process.
User Identifiers	The real user id identifies the user who is responsable for the running process. The effective user id may be used by a process to gain temporary privileges associated with a particular program; while that program is being executed as part of the process, the process operates with the effective user ID.
Process Identifiers	ID of this process; ID of parent process. These are set up when the process enters the Current state during the fork system call.
Event Descriptor	Valid when a process is in a sleeping stae; when the event occurs, the process is transferred to a ready to run state.
Priority	Used for process scheduling.
Signal	Enumerates signals sent to a process but not yet handled.
Timers	Include, process execution time, kernel resource utilization, and user-set timer used to send alarm signal to a process.
P_link	Pointer to the next link in the ready queue (valid if process is ready to execute).
Memory Status	Indicates whether process image is in main memory or swapped out. If it is in memory, this field also indicates whether it may be swapped out or is temporarily locked into main memory.

... 3.4 - Gerenciamento de Processos no UNIX SVR 4

✰ UNIX U Area:

Process Table Ponter	Indicates entry that corresponds to the U area.
User Identifiers	Real and effective user IDs. Used to determine user privileges.
Timers	Record time that the process (and its descendants) spent executing in user mode and in kernel mode.
Signal-Handler Array	For each type of signal defined in the system, it indicates how the process will react to receipt of the signal (exit, ignore, execute specified user function).
Control Terminal	Indicates login terminal for this process, if one exists.
Error Field	Records errors encountered during a system call.
Return Value	Contains the result of system calls.
I/O Parameters	Describes the amount of data to transfer, the address of the source (or target) data array in user space, and file offsets for I/O.
File Parameters	Current directory and current root describe the file system environment of the process.
User File Descriptor Table	Records the files the process has open.
Limit Fields	Restricts the size of the process and the size of a file it can write.
Permission Modes Fields	Masks mode settings on files the process creates.

... 3.4 - Gerenciamento de Processos no UNIX SVR 4

✰ Quando um processo executa uma chamada *fork(...)*, o sistema operacional executa, nesta ordem, as funções descritas abaixo:

❶ alocação de um *slot* na tabela de processos;

❷ atribuição de um identificador para o processo filho;

❸ cópia da imagem do processo pai, exceto memória compartilhada;

❹ incrementa o contador de qualquer arquivos mantidos pelo processo pai, para refletir que o processo criado também é dono daqueles arquivos;

❺ atribuição ao processo filho do estado **ready-to-run**;

❻ devolve o identificador do processo filho para o processo pai, e o valor 0 (*process identifier* ou pid 0) para o próprio processo filho.