UFRJ – IM - DCC

Sistemas Operacionais I

Unidade II - Processos

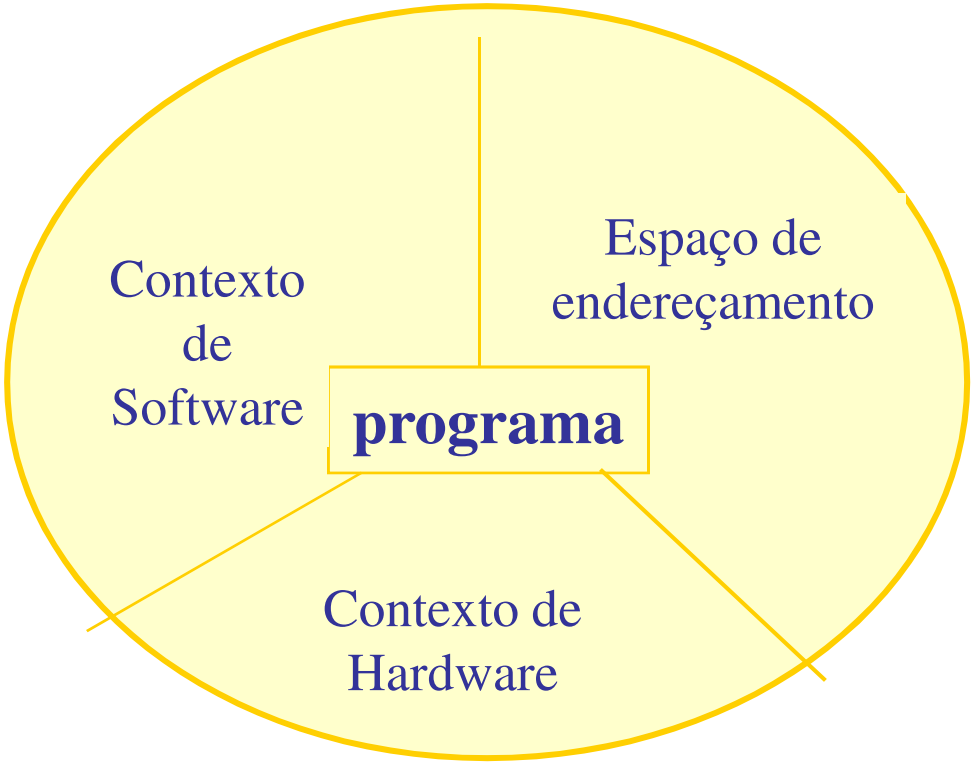
Organização da Unidade

- **Processos**
  - **Conceituação**
  - **Estruturas de Controle**
  - **Modos de execução do S.O.**
  - **Estados de um processo**
- **Threads**
- **Concorrência**
- ***Deadlock* e *Starvation***

Conceituação

Processo x Programa

- ✓ **Programa em execução**
- ✓ **Unidade de alocação do processador**



Estrutura de um Processo

- Contexto de Hardware
  - ✓ Armazena o conteúdo dos registradores da UCP. Fundamental para sistemas multitarefa.
- Contexto de Software
  - ✓ Especifica características e limites dos recursos que podem ser alocados pelo processo – definidos pelo administrador do sistema.
- Espaço de Endereçamento
  - ✓ Área de memória pertencente ao processo (instruções e dados).

Estrutura de um Processo

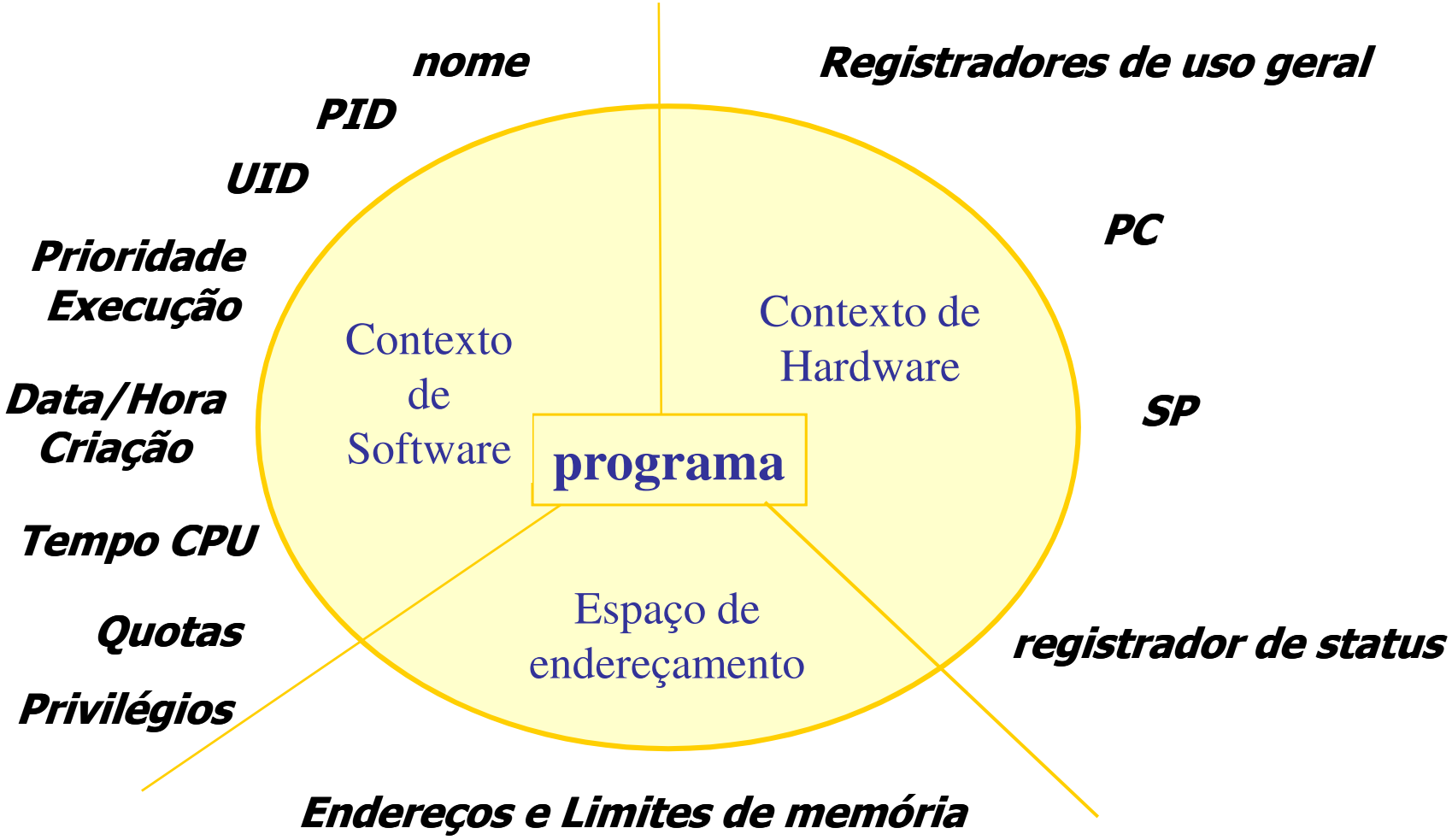
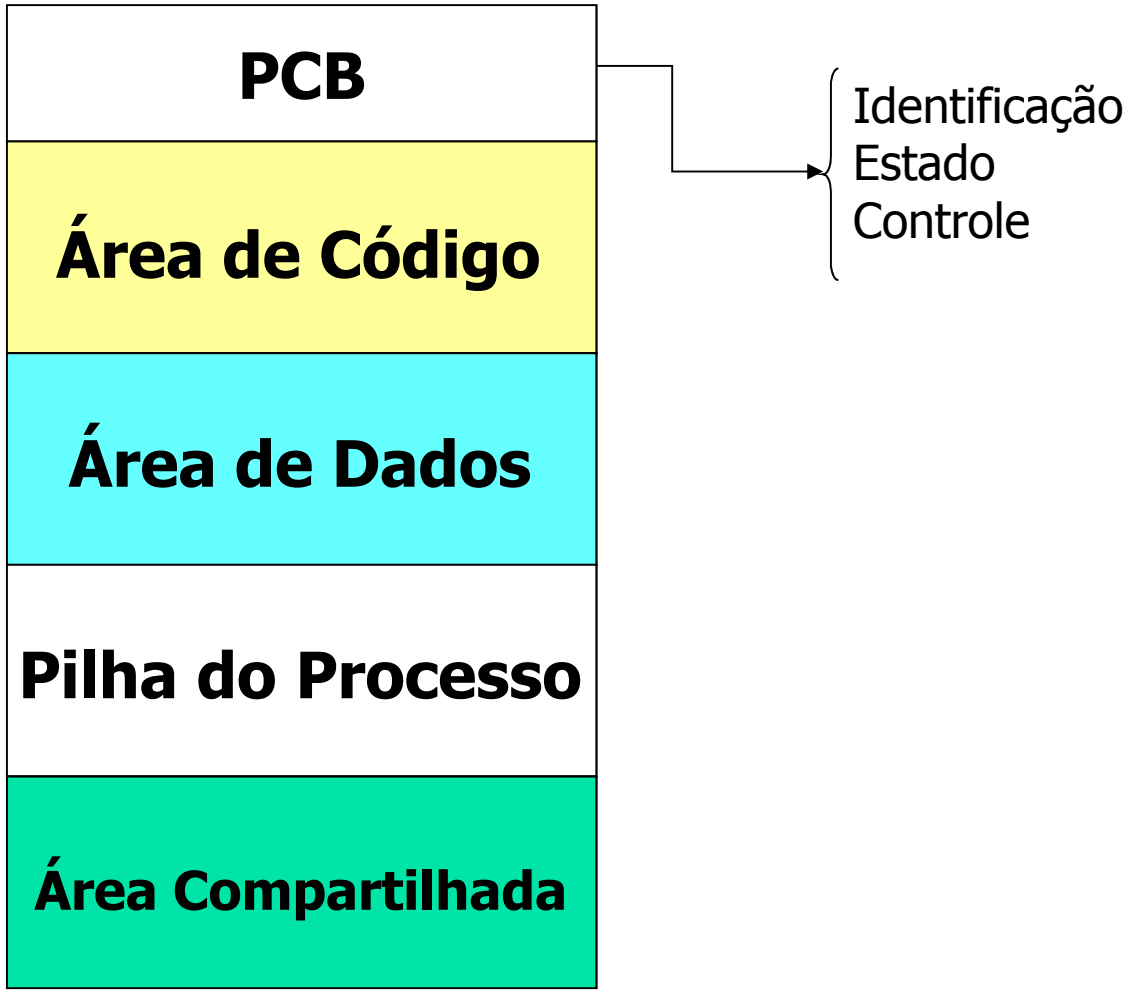
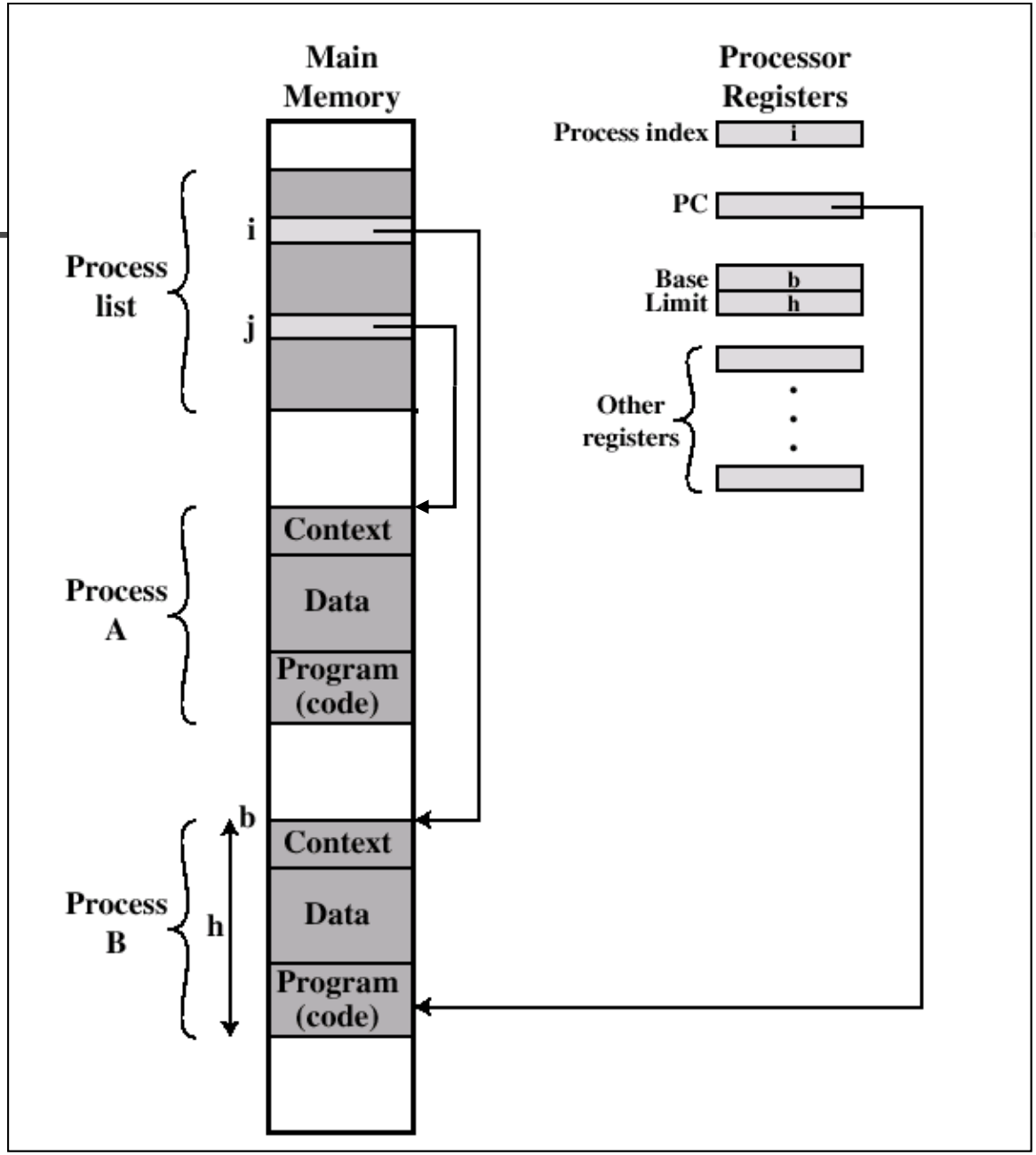


Imagem de um Processo



Processos


Main Memory
Processor Registers
Process index
i
PC
Base
Limit
b
h
Other registers
Process list
i
j
Process A
Context
Data
Program (code)
Process B
b
h
Context
Data
Program (code)

Estrutura de Controle

PCB => Process Control Block

- **Identificação**
- **Estado**
- **Controle**

identificação	estado
registrador SP	
registrador PC	
registradores de uso geral	
informações de escalonamento	
limites de memória	
privilégios	
relação de arquivos abertos	

Modos de Execução do S.O.

Modo usuário → instruções associadas ao uso não privilegiado

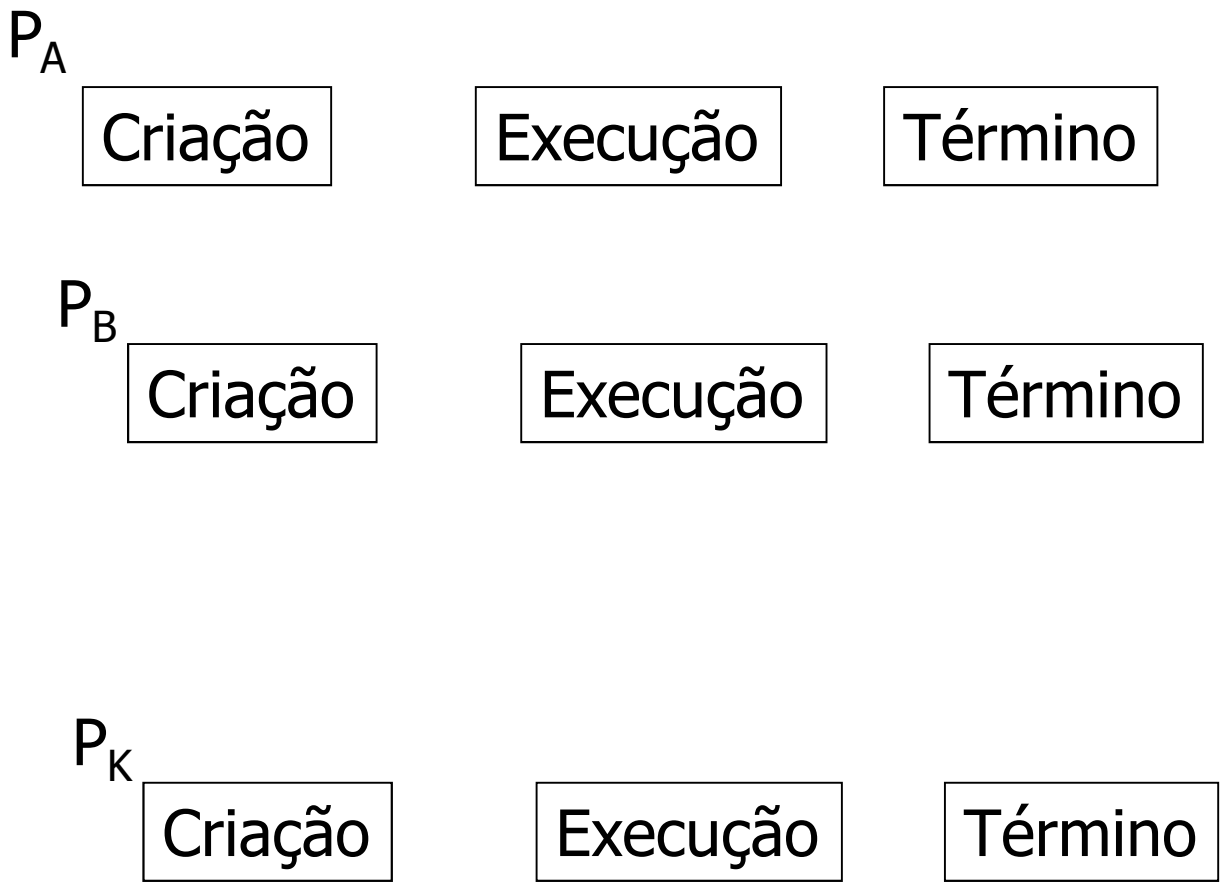
Modo kernel → instruções associadas ao uso privilegiado

Configuração:

Onde ?? Um bit do PSW

Quando ??? em resposta a determinados eventos

Ciclo de Vida de um Processo

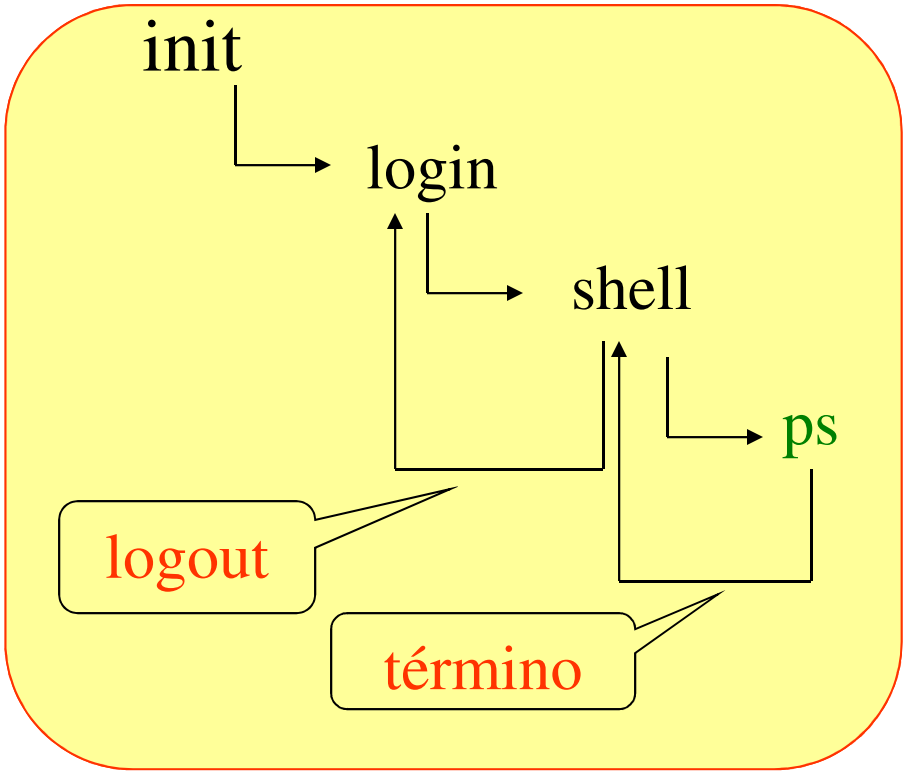


Criação de um processo

Quando ocorre?

- ✓ Nova tarefa em batch
- ✓ "Login" interativo
- ✓ Serviço do S.O
- ✓ Processo filho

Exemplo: Unix



Etapas de criação

- **Atribui um identificador único (PID)**
- **Aloca uma entrada na tabela de processos**
- **Aloca espaço para o processo**
- **Inicializa o PCB (Process Control Block)**
- **Coloca o processo na fila apropriada**
- **Cria estruturas auxiliares**

Como criar processos no Linux?

```
retorno = fork ();
if (retorno == 0)
{ /* o processo filho */
    execl ("/bin/ls", "ls", 0); /* troca programa
   */
    exit (1); /* não consegui executar pwd */
}
/* o processo pai */
if (retorno > 0)
{
    waitpid (-1, &status, 0);
}
else
{
    exit (1); /* não consegui dar o fork */
}
```

Execução

A execução concorrente de processos leva a situações que ensejam:

- Trocas de Contexto
- Trocas de Modo de Execução
- Formas de Execução do SO

Troca de contexto

Causas

- **Interrupção** : Reação a um evento assíncrono
- **Trap** : Associado a erro na execução de uma instrução
- **System Call** : Requisição explícita.

Ações Tomadas

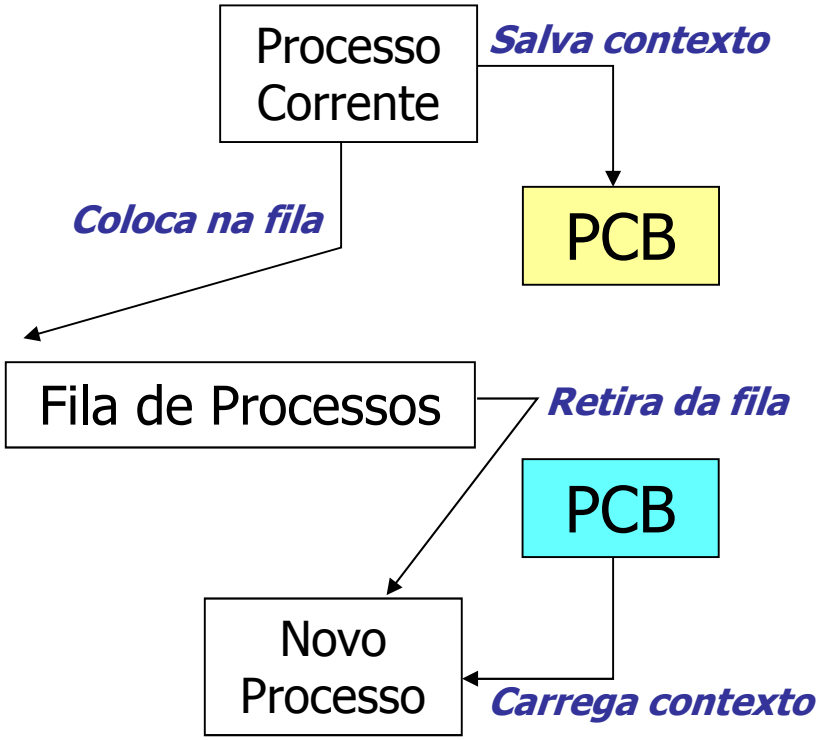
- ***Salva o estado do processador***
- ***Muda o estado do processo***
- ***Muda o processo para a fila apropriada***
- ***Seleciona o novo processo***
- ***Atualiza o PCB do novo processo***
- ***Modifica os mapeamentos de memória***
- ***Restaura o estado do processador***

Troca de Modo de Execução

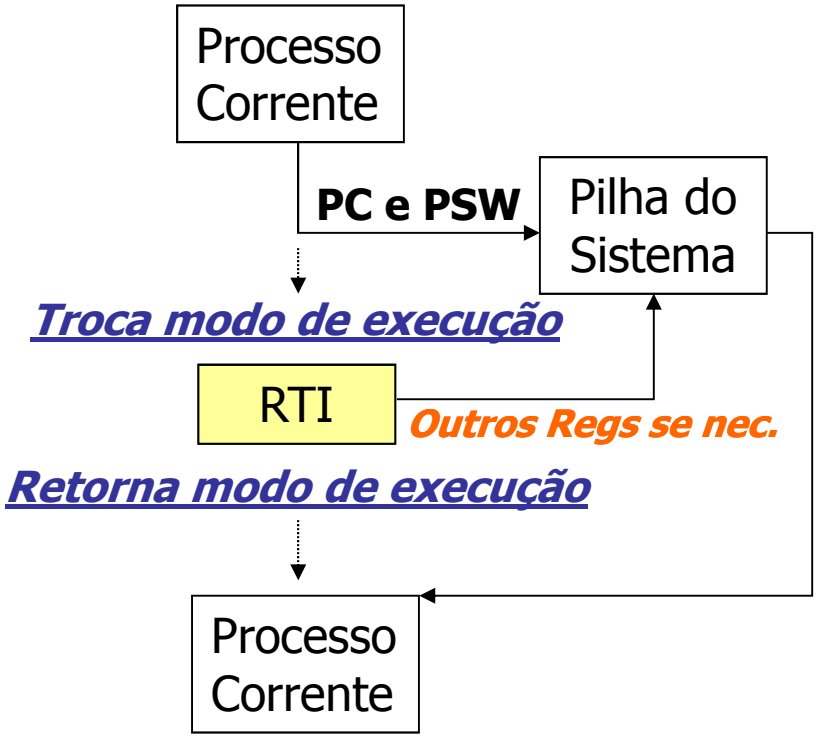
- **É uma troca menor e mais rápida que a troca de contexto;**
- **O estado do processo corrente não é alterado;**
- **Ocorre geralmente quando o processador ao final de um ciclo de instrução detecta a existência de uma interrupção pendente. Nestes casos o processador realiza os seguintes passos:**
  - **Salva o PC e a PSW do processo em execução na pilha;**
  - **Carrega o PC com o endereço inicial da rotina de interrupção;**
  - **Troca o modo de execução de usuário para kernel (privilegiado) para que instruções privilegiadas do tratador de interrupções possam ser executas.**

Resumo

Troca de Contexto



Troca de Modo de Execução



Formas de Execução do SO

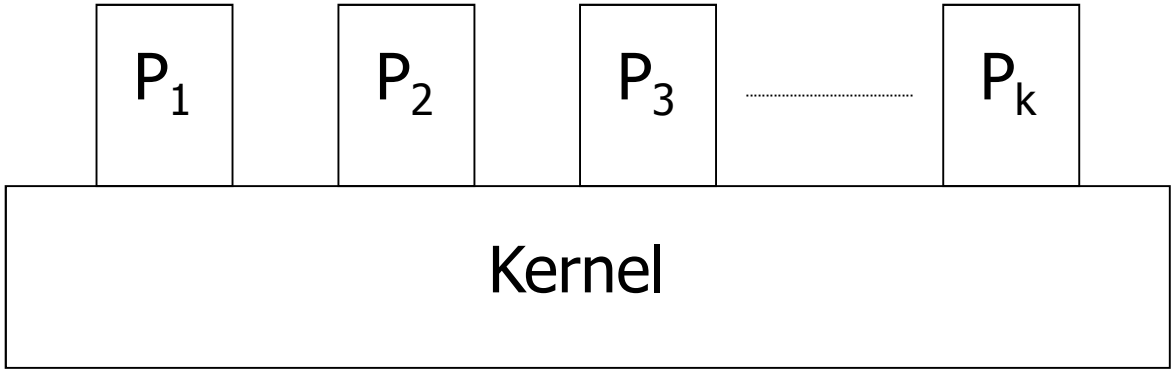
Sendo um processo, como as rotinas do SO devem ser executadas e controladas?

As principais abordagens encontradas na literatura são:

- *Como Kernel separado*
- *Dentro do processo usuário*
- *Como processos separados*

Formas de Execução do SO

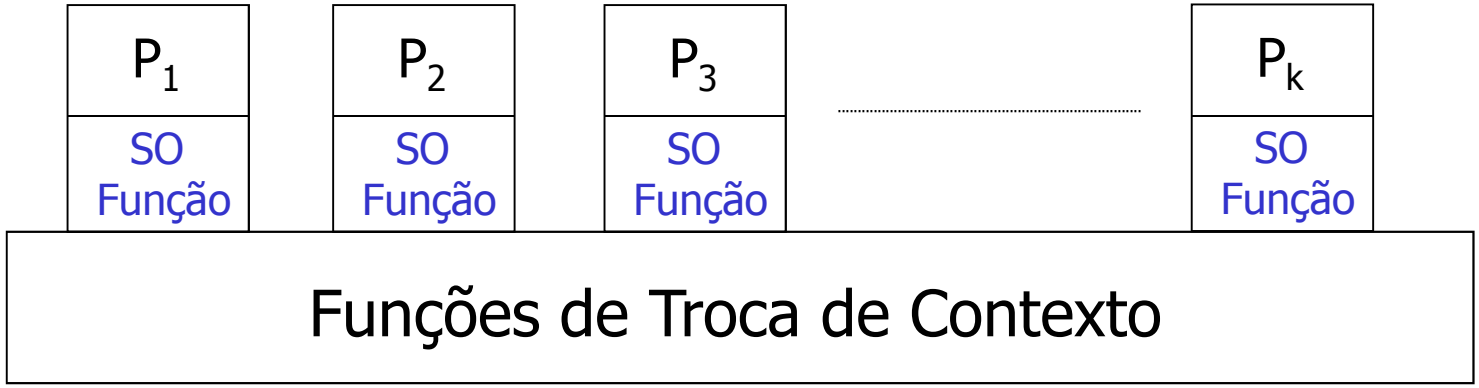
Como Kernel separado



(*) Nesta abordagem as rotinas do SO sempre são executadas como entidades separadas que operam no modo privilegiado e no espaço de endereçamento do Kernel.

Formas de Execução do SO

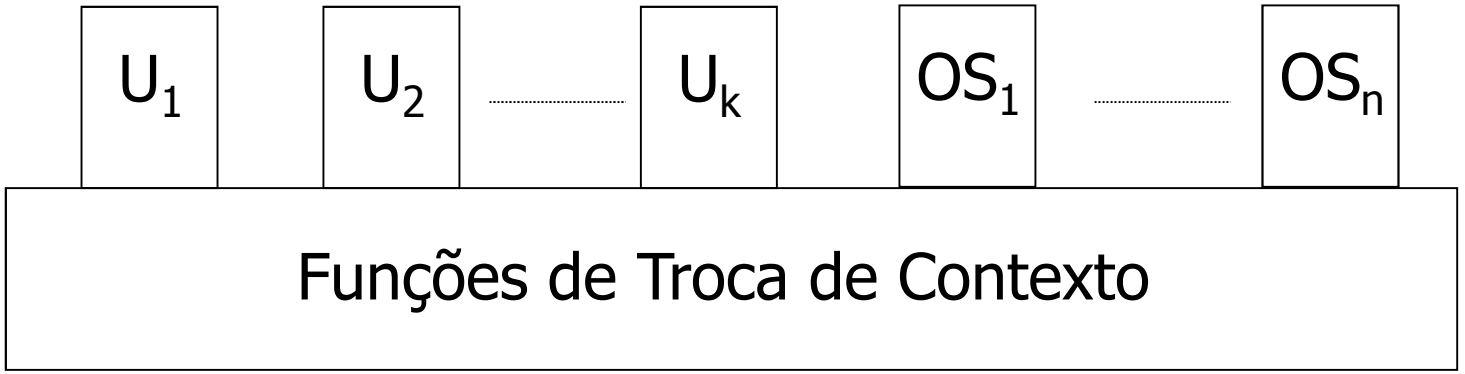
Dentro do processo usuário



(*) Nesta abordagem as rotinas do SO são executadas dentro dos processos usuários, que apenas mudam de modo de execução.

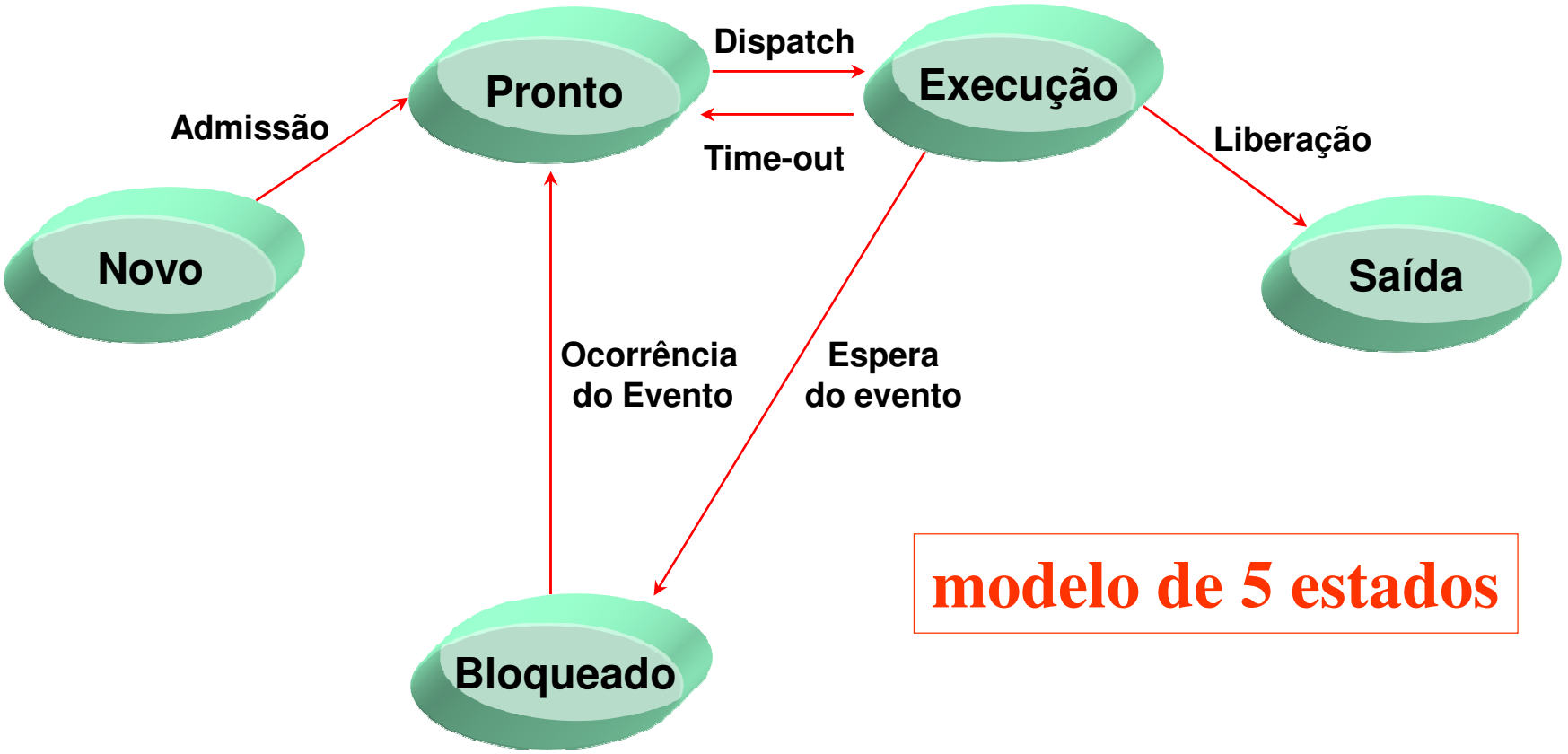
Formas de Execução do SO

Como processos separados



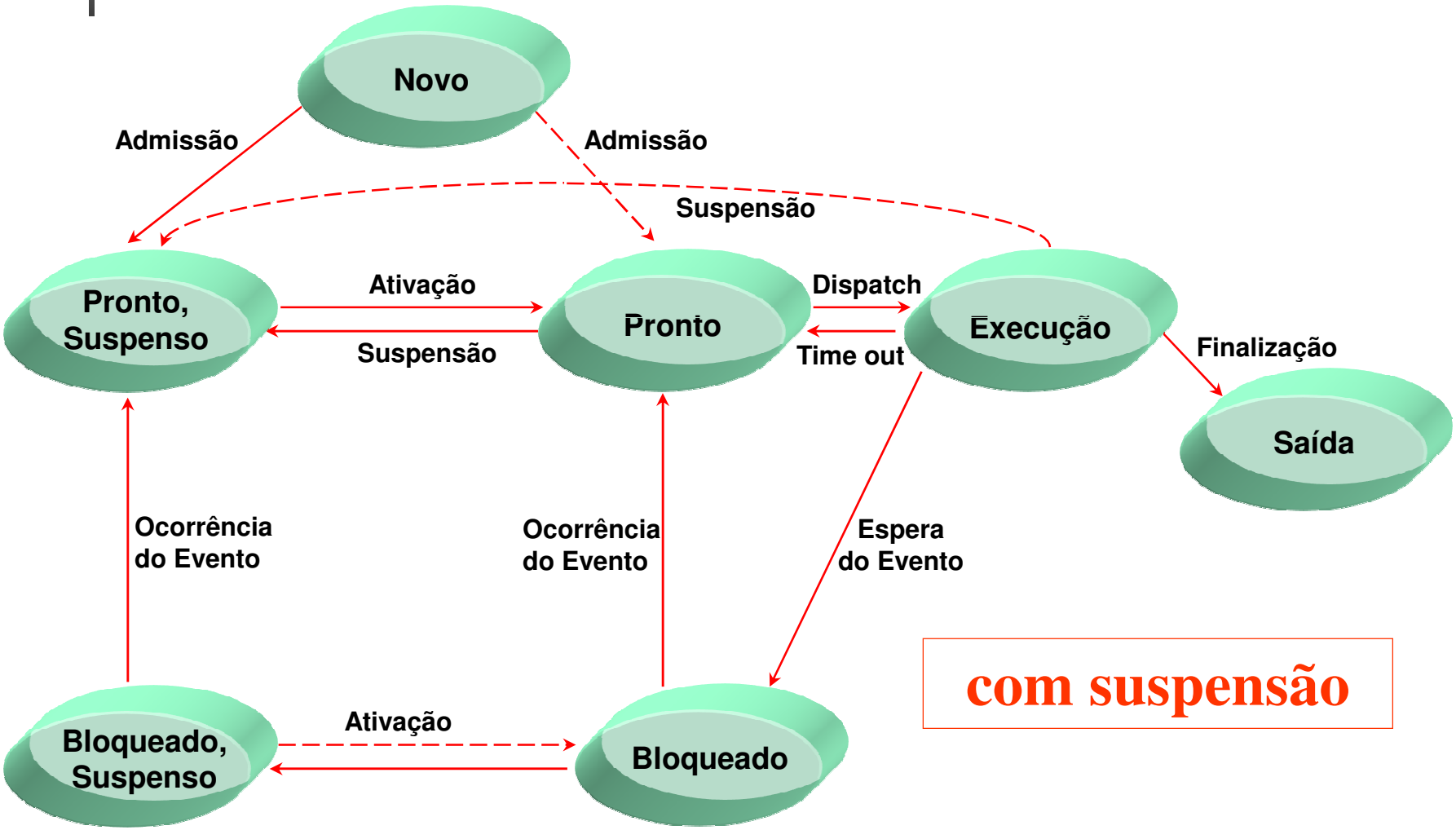
(*) Nesta abordagem as rotinas do SO são executadas como processos no modo usuário, trocando o modo de execução quando necessário.

Estados de um Processo



modelo de 5 estados

Estados de um Processo



Término de um Processo

- ✓ Término Normal
- ✓ Erros
  - Limite de tempo
  - Falta de recurso
  - Violação de acesso
  - ...
- ✓ Forçado

Motivo?