# Sistemas Operacionais I

Processos (parte 2)

Prof. Leandro Marzulo

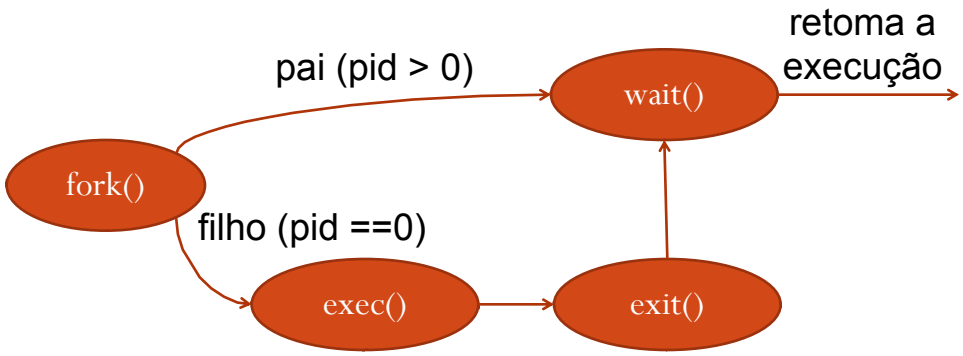


março de 2012

# Criação de processos

- Chamada fork()
- Retorna inteiro
  - < 0 → significa erro
  - == 0 → estamos no processo filho
  - > 0 → estamos no processo pai e o retorno é o PID do filho.
- Execução
  - O pai continua a ser executado concorrentemente com seus filhos
  - O pai espera até alguns de seus filhos ou todos serem encerrados
- Espaço de endereço
  - Cópia do pai (mesmo programa)
  - Um novo programa é carregado

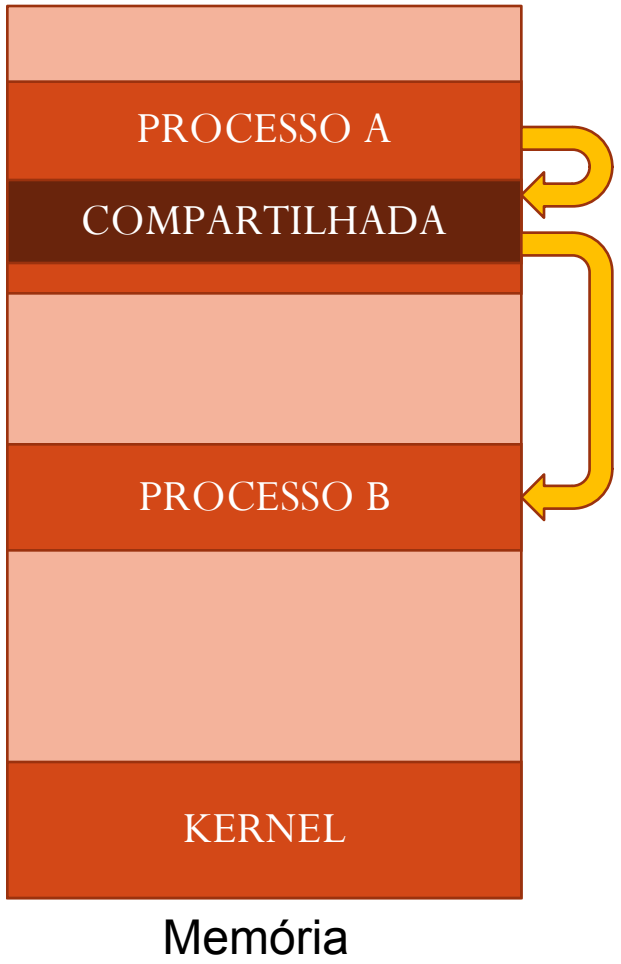
# Criação de Processos



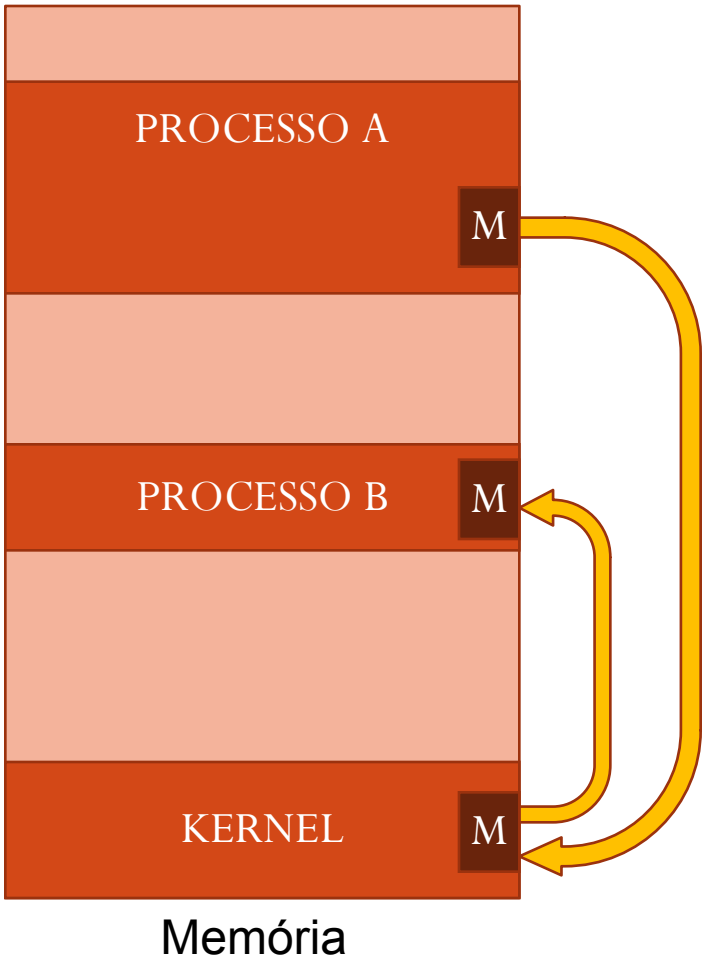

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>

int main()
{
    pid_t pid;
    pid = fork();

    if (pid < 0) { //erro
        fprintf(stderr, "Fork falhou!\n");
        return 1;
    }
    else if (pid == 0) { //processo filho
        execlp("/bin/ls", "ls", NULL);
    }
    else { //processo pai
        wait(NULL);
        printf("Filho terminou!\n");
    }
    return 0;
}
```

# Comunicação entre processos: Memória Compartilhada

- Um dos processos cria a região de memória compartilhada – shmget() – SHared Memory GET – retorna um identificador para o segmento
- Os processo querem ter acesso a esta região devem anexá-lo ao seu espaço de endereçamento – shmat() – SHared Memory ATtatch – precisa do identificador para o segmento
- Os processos se comunicam acessando normalmente esta região de memória
- Desanexa memória – shmdt() – SHared Memory DeTatch
- Destrói região compartilhada – shmctl() – SHared Memory ConTroL



# Comunicação entre processos: Troca de Mensagens



# Comunicação entre processos: Pipes

- Cria o pipe – pipe()
- Pai escreve em uma extremidade (e fecha a outra) – write()
- Filho lê da outra extremidade (e fecha a uma) – read()
- Também podem ser tratados como arquivos