

# LabSO

## Gerência de Processos

---

---

### AULA 3

Flávia Maristela ([flavia@flaviamaristela.com](mailto:flavia@flaviamaristela.com))  
Romildo Martins ([romildo@romildo.net](mailto:romildo@romildo.net))

## Processos

---

---

Porque eu preciso entender este assunto?

---

---

- Para entender como um computador consegue executar várias tarefas simultaneamente e **qual o impacto que isso pode ter em meus programas!**



## Os programas de ontem...

---

---

- Antigamente, os computadores eram máquinas dedicadas:
  - Possuíam apenas um usuário
  - Executavam apenas um programa por vez
    - Programas em execução tinham total controle dos recursos do computador

## E os programas de hoje!

---

---

- Hoje os computadores:
  - Executam vários programas simultaneamente
  - Podem ser usados por vários usuários
- Isso gerou a necessidade de compartilhar recursos...
- ... e por isso os programas foram divididos em unidades menores.

## Sobre os programas...

---

---

- Quando ligamos o computador, vários programas começam a ser executados.
  - Programas ativados pelo Sistema Operacional
  - Programas ativados pelo usuário
- Cada um destes programas possui vários **processos**.

## Processos (-- definição --)

---

---

- Definição:
    - Programa em execução
- Silberschatz, Tanenbaum*
- Processos são entidades independentes entre si, mas **concorrem** aos mesmos recursos do computador.

## Processos (-- estados --)

---

---

- **Novo**
  - O processo está sendo criado, ou seja, seu código está sendo carregado em memória, junto com suas bibliotecas;
  - As estruturas de dados do *kernel* estão sendo atualizadas para permitir sua execução.
- **Pronto**
  - Processo está em memória, pronto para ser executado, aguardando a disponibilidade do processador;
  - **IMPORTANTE**: Os processos “prontos” são organizados em uma fila cuja ordem é determinada por algoritmos de escalonamento.

## Processos (-- estados --)

---

---

- **Executando:**
  - Processo está executando suas instruções.
- **Bloqueado**
  - Processo não pode executar porque depende de recursos ainda não disponíveis (dados, algum tipo de sincronização, a liberação de algum recurso compartilhado);
  - Processo simplesmente espera o tempo passar (em estado de “*sleeping*”).
- **Terminado**
  - A execução do processo foi encerrada e ele pode ser removido da memória do sistema.

## Processos (-- transições --)

---

---

- **... → Novo**
  - um novo processo é criado e começa a ser preparado para executar.
- **Novo → Pronto**
  - o novo processo termina de ser carregado em memória, estando pronto para executar.
- **Pronto → Executando**
  - o processo é escolhido pelo escalonador para ser executado, entre os demais processos prontos.

## Processos (-- transições --)

---

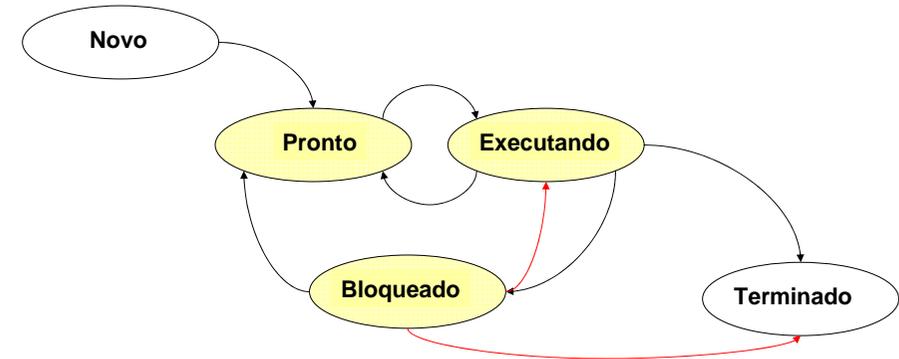
---

- **Executando → Pronto**
  - esta transição ocorre quando se esgota a fatia de tempo destinada ao processo (***quantum***);
  - Nesse momento o processo não precisa de outros recursos além do processador e por isso volta à fila de “pronto” para esperar novamente a disponibilidade do processador.
- **Executando → Terminada**
  - O processo encerra sua execução ou é abortado em consequência de algum erro (acesso inválido à memória, instrução ilegal, divisão por zero).
  - Em geral, o processo que deseja terminar avisa ao sistema operacional através de uma chamada de sistema.

## Processos (-- transições --)

- **Terminado** → ...
  - Quando terminado, um processo é removido da memória e seus registros e estruturas de controle no *kernel* são apagados.
- **Executando** → **Bloqueado**
  - caso o processo em execução solicite acesso a um recurso não disponível, ele abandona o processador e fica bloqueado até o recurso ficar disponível.
- **Bloqueado** → **Pronto**
  - quando o recurso solicitado pelo processo se torna disponível, ele pode então voltar ao estado de “pronto”.

## Processos (-- transições --)



## Processo (-- process control block --)

Representação do processo para o sistema operacional

pointer	process state
process number	
program counter	
registers	
memory limits	
list of open files	
⋮	

*Silberschatz, capítulo 2*

## Processo (-- process control block --)

- Informações associadas a cada processo:
  - Estado do processo
  - Valor do Contador de Programa
    - indica a próxima instrução a ser executada
  - Área para guardar valor dos registradores (dados)
  - Dados para gerenciamento da CPU (escalonamento)
  - Dados para gerenciamento de memória
  - Número do processos
  - Informações sobre E/S

## Processos (-- criação --)

- O que motiva a criação de um processo?
  - Inicialização do sistema operacional;
  - Inicialização de um programa;
  - Chamada de sistema;

## Processos (-- execução --)

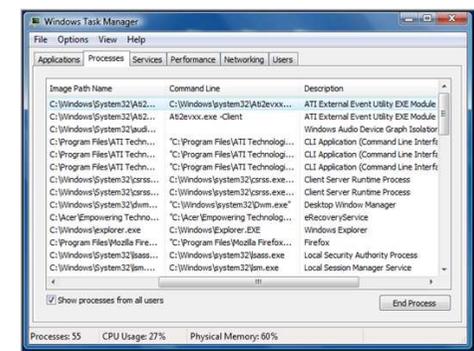
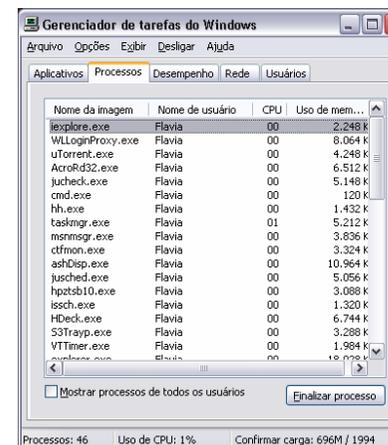
- Os processos podem executar de duas formas:
  - Em *FOREGROUND*
    - Processos que interagem com os usuários
  - EM *BACKGROUND*
    - Não associados a usuários
    - Possuem funções específicas

## Processos (-- finalização --)

- O que motiva a finalização de um processo?
  - Saída normal;
  - Saída com erro;
  - *Fatal Error* (involuntário);
  - Outro processo (involuntário)

## Visualizando os processos no Windows

### ■ Gerenciador de tarefas



# Visualizando os processos no Windows

■ Comando: *tasklist*

```
Prompt de comando
C:\Documents and Settings\Flavia>tasklist

Nome da imagem          Identi Nome da sessão  Sessão#  Uso de memór
-----
System Idle Process     0       Console        0          28 K
System                  4       Console        0        67.240 K
smss.exe                 604     Console        0          400 K
csrss.exe                652     Console        0         1.880 K
winlogon.exe             676     Console        0         1.528 K
services.exe            720     Console        0         3.508 K
lsass.exe                732     Console        0          1.436 K
svchost.exe              892     Console        0         5.264 K
svchost.exe              948     Console        0         4.504 K
svchost.exe             1444     Console        0        21.908 K
svchost.exe             1484     Console        0         3.368 K
svchost.exe             1544     Console        0         3.528 K
svchost.exe             1752     Console        0         4.416 K
aswupdsv.exe            1836     Console        0          260 K
ashServ.exe             1992     Console        0         38.820 K
augrsx.exe              2008     Console        0         79.352 K
spoolsv.exe             1148     Console        0         5.316 K
svchost.exe             1604     Console        0         4.204 K
vmware-authd.exe        1608     Console        0          3.764 K
vmtoolsd.exe            900     Console        0         3.976 K
```



# Visualizando os processos no Linux (-- ps - aux --)

```
Shell - Konsole
root@el[~]# ps -aux
Warning: bad ps syntax, perhaps a bogus '-?' See http://procps.sf.net/faq.html
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1   0.0  0.2  1948   400 ?        Ss   18:02   0:02 init [5]
root         2   0.0  0.0   0   0 ?        RN   18:02   0:00 [ksoftirqd/0]
root         3   0.0  0.0   0   0 ?        S   18:02   0:00 [watchdog/0]
root         4   0.0  0.0   0   0 ?        S<   18:02   0:00 [events/0]
root         5   0.0  0.0   0   0 ?        S<   18:02   0:00 [khelper]
root         6   0.0  0.0   0   0 ?        S<   18:02   0:00 [kthread]
root         9   0.0  0.0   0   0 ?        S<   18:02   0:00 [kblockd/0]
root        10   0.0  0.0   0   0 ?        S<   18:02   0:00 [kacpid]
root        65   0.0  0.0   0   0 ?        S<   18:02   0:00 [kqueue/0]
root        67   0.0  0.0   0   0 ?        S<   18:02   0:00 [kseriod]
root       124   0.0  0.0   0   0 ?        S   18:02   0:00 [pdflush]
root       125   0.0  0.0   0   0 ?        S   18:02   0:00 [pdflush]
root       126   0.0  0.0   0   0 ?        S<   18:02   0:00 [kswapd0]
root       127   0.0  0.0   0   0 ?        S<   18:02   0:00 [aio/0]
root       128   0.0  0.0   0   0 ?        S<   18:02   0:00 [xfslogd/0]
root       129   0.0  0.0   0   0 ?        S<   18:02   0:00 [xfsdatad/0]
root       265   0.0  0.0   0   0 ?        S<   18:02   0:00 [ata/0]
root       266   0.0  0.0   0   0 ?        S<   18:02   0:00 [ata aux]
root       296   0.0  0.0   0   0 ?        S<   18:02   0:00 [kcryptd/0]
root       297   0.0  0.0   0   0 ?        S<   18:02   0:00 [kmpathd/0]
root       298   0.0  0.0   0   0 ?        S<   18:02   0:00 [kmlrord]
root       301   0.0  0.0   0   0 ?        S<   18:02   0:00 [kded]
root       351   0.0  0.0   0   0 ?        S<   18:02   0:00 [khubd]
root       725   0.0  0.5  2732 1140 ?        Ss+  18:02   0:01 udevd --daemon
root      1026   0.0  0.0   0   0 ?        S<   18:02   0:00 [kpmousead]
root      1093   0.0  0.0   0   0 ?        S<   18:03   0:00 [kgameportd]
root     1290   0.0  0.2  1500  396 ?        Ss   18:03   0:00 /usr/sbin/acpid -c /etc/acpi/event
root     1756   0.0  0.2  2804  428 ?        Ss   18:03   0:00 /usr/bin/kdm
root     1759   3.1 15.8 36496 30112 tty7    Ss+  18:03   1:55 /usr/X11R6/bin/X -dpi 75 -nolisten
root     1760   0.0  0.3  3712   640 ?        S   18:03   0:00 -i
kurumin    2000   0.0  1.0  4468 1972 ?        Ss   18:03   0:00 /bin/sh /etc/kde3/kdm/Xsession /us
```



# Visualizando os processos no Linux (-- top --)

```
Shell - Konsole
root@el[~]# top
top - 18:51:52 up 49 min, 0 users, load average: 0.58, 0.51, 0.46
Tasks: 85 total, 2 running, 83 sleeping, 0 stopped, 0 zombie
Cpu(s): 4.7%us, 8.3%sy, 0.0%ni, 84.7%id, 0.0%wa, 2.3%hi, 0.0%si, 0.0%st
Mem: 190064k total, 181928k used, 8136k free, 4532k buffers
Swap: 0k total, 0k used, 0k free, 44352k cached

  PID USER      PR  NI  VIRT  RES  SHR  %CPU  %MEM    TIME+  COMMAND
 1759 root        15   0 59532 33m 2428 S  6.3 18.3  1:19.46 Xorg
12439 kurumin    15   0 28812 14m 12m S  1.7  8.0   0:01.38 ksnapshot
 2079 kurumin    21   0 33620 13m 10m S  1.3  7.2   0:51.85 rded
 7251 kurumin    15   0 61828 24m 8148 S  1.0 13.4   0:36.04 konqueror
 5123 kurumin    15   0 31188 10m 6792 S  0.3  5.5   0:05.23 console
   1 root         15   0 1948 512 420 S  0.0  0.3   0:02.07 init
   2 root         15   0 0 0 S  0.0  0.0   0:00.01 ksoftirqd/0
   3 root         17   0 0 0 S  0.0  0.0   0:00.00 watchdog/0
   4 root         10  -5 0 0 S  0.0  0.0   0:00.00 events/0
   5 root         10  -5 0 0 S  0.0  0.0   0:00.04 khelper
   6 root         10  -5 0 0 S  0.0  0.0   0:00.00 kthread
   9 root         10  -5 0 0 S  0.0  0.0   0:00.01 kblockd/0
  10 root         20  -5 0 0 S  0.0  0.0   0:00.00 kacpid
  65 root         20  -5 0 0 S  0.0  0.0   0:00.00 kqueue/0
  67 root         10  -5 0 0 S  0.0  0.0   0:00.02 kseriod
 124 root         15   0 0 0 S  0.0  0.0   0:00.02 pdflush
 125 root         15   0 0 0 S  0.0  0.0   0:00.04 pdflush
 126 root         10  -5 0 0 S  0.0  0.0   0:00.22 kswapd0
 127 root         20  -5 0 0 S  0.0  0.0   0:00.00 aio/0
 128 root         20  -5 0 0 S  0.0  0.0   0:00.00 xfslogd/0
 129 root         20  -5 0 0 S  0.0  0.0   0:00.00 xfsdatad/0
 265 root         11  -5 0 0 S  0.0  0.0   0:00.00 ata/0
 266 root         11  -5 0 0 S  0.0  0.0   0:00.00 ata_aux
 296 root         11  -5 0 0 S  0.0  0.0   0:00.00 kcryptd/0
 297 root         11  -5 0 0 S  0.0  0.0   0:00.00 kmpathd/0
 298 root         11  -5 0 0 S  0.0  0.0   0:00.00 kmlrord
 301 root         10  -5 0 0 S  0.0  0.0   0:00.00 kded
 351 root         10  -5 0 0 S  0.0  0.0   0:00.00 khubd
 725 root         21  -4 2732 1184 364 S  0.0  0.6   0:01.05 udevd
```



# Visualizando os processos no Linux

■ *ps -aux*

- É uma fotografia dos processos em execução
- Exibe todos os processos e seus usuários

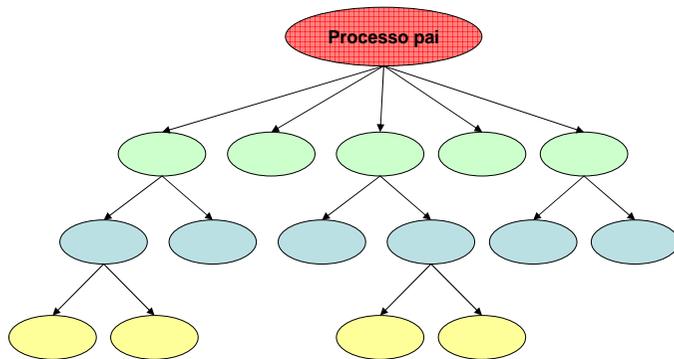
■ *top*

- Fornece uma visão dinâmica dos processos em execução



## Processos (-- hierarquia --)

- Num sistema operacional, cada processo pode criar outros, através das chamadas de sistema.
- Isso dá origem a árvore de processos



## Árvore de processos no Windows

Process	Description	Image Path	Life Time	Owner
Idle (0)	Idle			AUTORIDAD
System (4)	System			USUARIO\F
Explorer.exe (2352)	Windows Explorer	C:\WINDOWS\Explorer.exe		USUARIO\F
VTTimer.exe (2640)	Windows Explorer	C:\WINDOWS\system32\VTTimer.exe		USUARIO\F
S3trapp.exe (2656)	S3trapp	C:\WINDOWS\system32\S3trapp.exe		USUARIO\F
i3conifl (3236)	HDcheck.MFC Appl.	C:\Arquivos de programas\VAW\Audiot\HDA\check\HDcheck.exe		USUARIO\F
HDcheck.exe (2676)	HDcheck.MFC Appl.	C:\Arquivos de programas\VAW\Audiot\HDA\check\HDcheck.exe		USUARIO\F
InstatShield Upda...	InstatShield Upda...	C:\Arquivos de programas\Arquivos comuns\InstatShield\UpdateSe...		USUARIO\F
hpqbtst10.exe (2716)	InstatShield Upda...	C:\WINDOWS\system32\upool\drivers\hw3268\3\hpqbtst10.exe		USUARIO\F
jpsched.exe (2724)	Java(TM) Platform...	C:\Arquivos de programas\Java\jre1.7.0\bin\jpsched.exe		USUARIO\F
jcheck.exe (3688)	Java(TM) Update...	C:\Arquivos de programas\Java\jre1.7.0\bin\jcheck.exe		USUARIO\F
ahDisp.exe (2886)	avast! service GU...	C:\ARQUIVOS DE PROGRAMAS\AVAST\AVAST4\ashDisp.exe		USUARIO\F
ofmon.exe (2900)	CTF Loader	C:\WINDOWS\system32\ctfmon.exe		USUARIO\F
mmsmsg.exe (2964)	Windows Live Me...	C:\Arquivos de programas\Windows Live\Messenger\mmsmsg.exe		USUARIO\F
explorer.exe (4044)	Internet Explorer	C:\Arquivos de programas\Internet Explorer\explorer.exe		USUARIO\F
AcroRd32.exe (3740)	Adobe Reader 8.1	C:\Arquivos de programas\Adobe\Reader 8.0\Reader\AcroRd32.exe		USUARIO\F
ulorent.exe (3940)	ulorent	C:\Arquivos de programas\ulorent\ulorent.exe		USUARIO\F
iis.exe (3936)	Microsoft® HTML	C:\WINDOWS\system32\iis.exe		USUARIO\F
POWERPNT.exe (436)	Microsoft Office P...	C:\Arquivos de programas\Microsoft Office\OFFICE11\FPOWERPNT...		USUARIO\F
avgsrv.exe (640)	AVG Scanning Co...	C:\Arquivos de programas\AVG\AVG8\avgsvr.exe		USUARIO\F
explorer.exe (2364)	Internet Explorer	C:\Arquivos de programas\Internet Explorer\explorer.exe		USUARIO\F
Process Monitor	Process Monitor	C:\Documents and Settings\Flavio\Meus documentos\programas\UF...		USUARIO\F
verity.exe (2748)	Verity Class ID	C:\WINDOWS\system32\verity.exe		USUARIO\F
veecld.exe (3324)	Verity Class ID	C:\WINDOWS\system32\veecld.exe		USUARIO\F

Description: Windows Live Messenger  
Company: Microsoft Corporation  
Path: C:\Arquivos de programas\Windows Live\Messenger\mmsmsg.exe  
Command: "C:\Arquivos de programas\Windows Live\Messenger\mmsmsg.exe" /background  
User: USUARIO\Flavio  
PID: 2964 Started: 2009-08-08 11:57



## Árvore de processos no Linux

- `ps -aux`
  - Exibe a árvore de processos



# Dúvidas?



# Exercícios I

---

---

1. O que são processos?
2. O que motivou sua criação?
3. Como o sistema operacional enxerga um processo?
4. Quais os atributos de um processo para o sistema operacional?
5. Quais os estados de um processo?
6. Quais as possíveis transições entre os processos?
7. O que é a árvore de processos?
8. O que pode motivar a criação ou finalização de um processo?
9. Como os processos podem executar?

## Threads

---

---

## THREADS

---

---

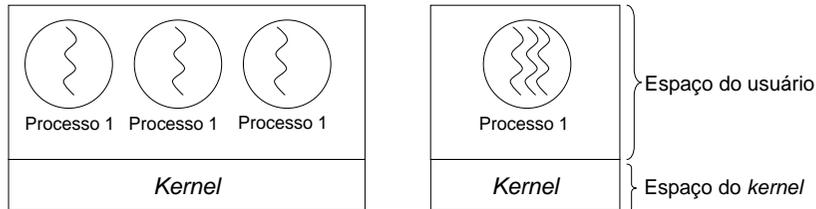


### ■ Motivação:

- A necessidade de compartilhar diferentes recursos do computador deu origem a *PROGRAMAÇÃO CONCORRENTE*.
- Neste cenário, um programa que tinha vários processos com um único fluxo de execução passou a ter vários processos.
- Cada processo possuía um ou mais fluxos de execução.

# THREADS

- *Threads* representam uma abstração para uma nova subdivisão necessária para os processos.



# THREADS

- Definição:
  - “Entidades escalonadas para execução”  
*Tanenbaum*
  - “Fluxo de execução dentro de um processo”  
*Rômulo Oliveira*
  - “Unidade básica de utilização da CPU”  
*Silbershatz*
- Assim como os processos, as *threads* também possuem estados.

# THREADS

- *Multithreading*
  - Termo usado para caracterizar um processo com várias *threads*.
  - Sistema **multithread** executa as *threads* tão rapidamente, que passa ao usuário a impressão de que as mesmas estão sendo executadas em paralelo.
  - O termo também está ligado a dispositivos de hardware que permitem a execução de várias *threads*.



**PARA PENSAR!** Qual é a diferença entre os seguintes termos:

- Paralelismo
- Pseudo-paralelismo
- Multiprogramação
- Multithreading*
- Time sharing*

# THREADS

- *Threads* compartilham os recursos de um processo;
- *Threads* de um mesmo processo não são independentes entre si
- Em sistemas *multithread*, normalmente cada processo inicia com apenas uma *thread*
  - Esta *thread* tem a capacidade de criar novas *threads*

## THREADS

---

---

- *Threads* não representam a solução para todos os problemas:
  - Se um processo é duplicado, ele deve manter todas as *threads* do processo pai?
  - Se uma *thread* estava bloqueada no momento da cópia de um processo, a *thread* filha também vai estar?
  - Quando um dado é útil para uma *thread*, quem vai receber uma cópia, apenas o processo pai? O processo filho também deve receber?

## THREADS

---

---

- *Threads* podem ser gerenciadas em dois níveis:
  - Nível do usuário
  - Nível do *kernel*

## THREADS (-- nível do usuário --)

---

---

- *Kernel* do sistema operacional não tem conhecimento sobre tais *threads*.
- Sistema operacional enxerga apenas um único processo com uma única *thread*.

## THREADS (-- nível do kernel --)

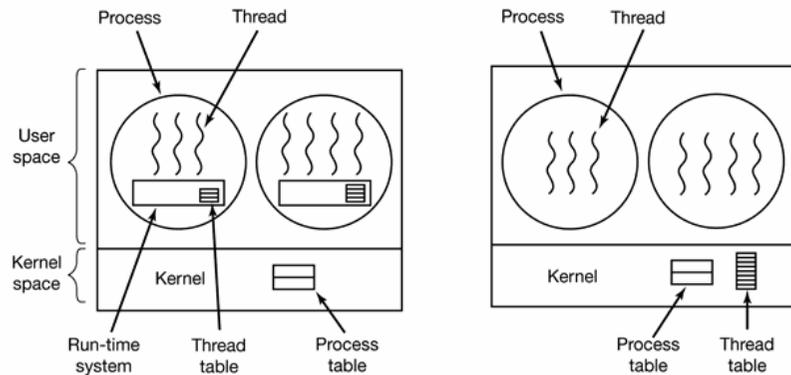
---

---

- *Kernel* do sistema operacional controla todas as operações entre *threads*:
  - *Create*
  - *Terminate*;
  - *Join*
  - *Yield*
  - *Resource sharing* (compartilhamento de recursos)

# THREADS

(-- nível usuário vs. nível kernel --)



## Dúvidas?



## Exercícios I

1. O que são *threads*?
2. Porque as *threads* foram criadas?
3. Explique as operações entre *threads*.
4. Explique como as *threads* são criadas.
5. Explique como as *threads* podem ser executadas.
6. Quais os estados de uma *thread*?

## Comunicação entre processos

## Comunicação entre processos (-- motivação --)

---

---

- Processos em execução no sistema operacional podem ser:
  - Independentes:
    - Quando não podem ser afetados pela execução de outro processo
  - Cooperantes
    - Quando podem ser afetados pela execução de outro processo
- Já sabendo que compartilhamento causa problemas, é mais fácil criar processos independentes!!

## Comunicação entre processos (-- motivação --)

---

---

- Entretanto, é extremamente desejável criar um ambiente com processos cooperantes!
- Porque?
  - Compartilhamento de informações
  - Aumento da velocidade de computação
  - Modularidade
  - Dar suporte a execução de várias tarefas
- Processos cooperantes requerem comunicação entre processos (*Interprocess communication – IPC*)

## Comunicação entre processos (-- definição --)

---

---

- Mecanismo que permite aos processos trocarem dados ou informações.
- Comunicação entre processos não usa interrupção!
- Frequentemente é feita de duas formas:
  - Troca de mensagens
  - Compartilhamento de memória

## Comunicação entre processos (-- troca de mensagens --)

---

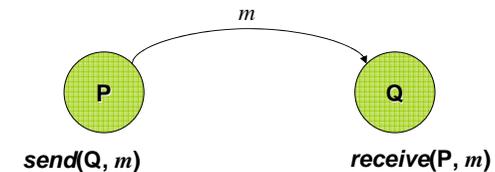
---

- Se pensarmos numa arquitetura centralizada, os processos estão na mesma máquina.
  - Diferentes processos têm acesso aos mesmos recursos.
- O que acontece se a arquitetura do sistema for distribuída? (um *chat*, por exemplo)
- Como os processos podem se comunicar?

## Comunicação entre processos (-- troca de mensagens --)

- Processos podem se comunicar por troca de mensagens.
  - Frequentemente quando estão em diferentes máquinas e precisam compartilhar dados
- A troca de mensagens é feita baseada em duas **primitivas**:
  - `send()`
  - `receive()`
- Mensagens podem ter tamanho fixo ou variável
- Se dois processos precisam se comunicar, deve haver um **link** entre eles.

## Comunicação entre processos (-- troca de mensagens --)

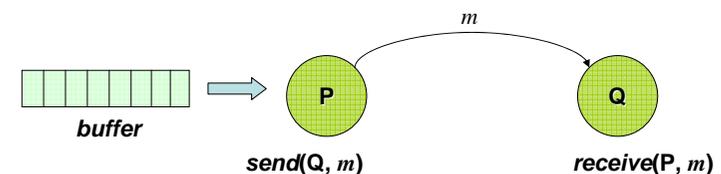


## Comunicação entre processos (-- troca de mensagens --)

- Troca de mensagens por sincronização:
  - Blocking send: processo que envia a mensagem fica bloqueado até a confirmação do recebimento
  - Nonblocking send: processo envia a mensagem e vai executar a próxima instrução
  - Blocking receive: receptor fica bloqueado até que a mensagem esteja disponível
  - Nonblocking receive: o receptor devolve uma mensagem válida ou nula.

## Comunicação entre processos (-- troca de mensagens --)

- Troca de mensagens por bufferização:
  - *Zero capacity*
  - *Bounded-capacity*
  - *Unbounded-capacity*



## Comunicação entre processos (-- compartilhamento de memória --)

---

- Processos devem definir uma área de memória que será compartilhada;
- Por padrão o sistema operacional não permite que um processo acesse outro processo!
- Como resolver???

## Comunicação entre processos (-- compartilhamento de memória --)

---

- Caso dois processos desejem compartilhar memória, ambos precisam assumir as consequências de não considerar as restrições do sistema operacional

## Comunicação entre processos (-- compartilhamento de memória --)

---

- Processos trocam informações através de leituras e escritas numa área compartilhada;
- O sistema operacional não controla esta operação!
- O que os processos precisam garantir??

Para pensar um pouco...

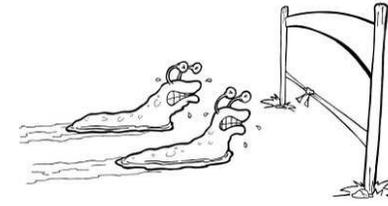
---

*O que acontece quando dois processos querem escrever na mesma área de memória no mesmo instante?*

## Comunicação entre processos (-- Race condition --)

- Em alguns sistemas operacionais, processos cooperantes frequentemente compartilham algum dispositivo de armazenamento.
  - Arquivos
  - Memória
  - Disco

## Comunicação entre processos (-- Race condition --)



- Dois processos podem tentar ler ou escrever dados num espaço compartilhado, e o resultado final depende de quem está executando naquele momento.

## Comunicação entre processos (-- Race condition: exemplo ilustrativo --)

- Um exemplo ilustrativo:
  - Suponha duas *threads*, que alteram o valor da variável  $x$

$T_1: x := x + 1$   
 $T_2: x := x + 2$

$T_1: x := x + 1$   
 $T_2: x := x * 2$

Considere  $x = 2$

Considere  $x = 2$

$T_1 \rightarrow T_2: x = 5$   
 $T_2 \rightarrow T_1: x = 5$

$T_1 \rightarrow T_2: x = 6$   
 $T_2 \rightarrow T_1: x = 5$

## Comunicação entre processos (-- Race condition: exemplo clássico --)

- PRODUTOR vs. CONSUMIDOR
  - Dois processos compartilham um buffer de tamanho fixo. Um deles (processo **produtor**) coloca dados no buffer. O outro (processo **consumidor**), remove estes dados.
  - Você consegue ver o problema?