



Sistemas Operativos

Capítulo 3



Patrício Domingues
ESTG/IPLeiria, 2019

✓ Programa

- É o ficheiro executável (.exe no Windows)

✓ Processo

- Um programa em execução
- Vários processos podem estar a correr o mesmo programa chrome.exe (ver figura)
- Cada processo contém
 - **Contador programa (PC)**
 - **Identificador, dono, atributos segurança**
 - **etc.**

The screenshot shows the Windows Task Manager application with the 'Processes' tab selected. A red rectangular box highlights a list of 20 instances of 'Brave Browser'. The instances are listed with their names, status, CPU usage, memory usage, disk usage, network usage, and GPU usage. A tooltip is visible over the 'Network' column header, displaying 'Network utilization on the current primary network'.

Name	Status	3% CPU	38% Memory	1% Disk	0% Network	0% GPU
Microsoft PowerPoint (32 bit)		0%	71.4 MB	0 MB/s	0 Mbps	0%
Microsoft Excel (32 bit) (2)		0%	51.1 MB	0 MB/s	0 Mbps	0%
Keepass		0%	8.7 MB	0 MB/s	0 Mbps	0%
Brave Browser (27)		0.6%	2,234.8 MB	0 MB/s	0.1 Mbps	0%
Brave Browser		0.2%	103.2 MB	0 MB/s	0.1 Mbps	0%
Brave Browser		0%	0.4 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	665.1 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	17.1 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	6.2 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	7.0 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	89.7 MB	0 MB/s	0 Mbps	0%
Brave Browser		0.1%	262.5 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	2.1 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	2.2 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	36.8 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	19.0 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	26.9 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	18.3 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	7.2 MB	0 MB/s	0 Mbps	0%
Brave Browser		0.1%	10.0 MB	0 MB/s	0 Mbps	0%
Brave Browser		0.1%	280.2 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	108.2 MB	0 MB/s	0 Mbps	0%
Brave Browser		0%	176.5 MB	0 MB/s	0 Mbps	0%

Comando “top” do linux

142 processos

uptime

Carga média do sistema

```

user@ubuntu: /bin
File Edit Tabs Help
top - 15:50:52 up 1 day, 9:44, 11 users, load average: 0.70, 0.45, 0.50
Tasks: 142 total, 1 running, 141 sleeping, 0 stopped, 0 zombie
%Cpu(s): 3.5 us, 0.7 sy, 0.0 ni, 95.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 1026012 total, 916316 used, 109696 free, 121464 buffers
KiB Swap: 0 total, 0 used, 0 free. 518944 cached Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM     TIME+ COMMAND
 1450 root        20   0  216612  41560 10060  S   3.9   4.1   6:23.45 Xorg
 2531 user        20   0  213720  24964 18468  S   0.3   2.4   5:33.43 vmtoolsd
 2518 user        20   0  220292  21220 13288  S   0.0   2.1   1:20.05 pcmanfm
 2510 user        20   0  208644  18860 11116  S   0.0   1.8   0:42.46 lxpanel
 2463 user        20   0  123992  16992  9664  S   0.3   1.7   0:15.91 ibus-ui-gtk3
 7731 user        20   0  187512  15712 10864  S   1.6   1.5   0:14.52 lxterminal
 2505 user        20   0   34120  11848  7240  S   0.0   1.2   0:26.95 openbox
 2441 user        20   0   48472  10488  5148  S   0.0   1.0   0:02.08 lxsession
 1247 root        20   0   52116   9796  4676  S   0.0   1.0   0:23.12 NetworkManager
 2429 user        20   0   45764   8640  2896  S   0.3   0.8   1:54.88 ibus-daemon
 9728 user        20   0   49172   8580  5668  S   0.0   0.8   0:00.18 vim
 9750 user        20   0   49172   8580  5668  S   0.0   0.8   0:00.18 vim
 9788 user        20   0   49172   8580  5668  S   0.0   0.8   0:00.17 vim
 9821 user        20   0   49172   8580  5668  S   0.0   0.8   0:00.16 vim
 9840 user        20   0   49172   8580  5668  S   0.0   0.8   0:00.27 vim
 9864 user        20   0   49172   8580  5668  S   0.0   0.8   0:00.18 vim
 9730 user        20   0   49172   8576  5668  S   0.0   0.8   0:00.14 vim
 9769 user        20   0   49172   8576  5668  S   0.0   0.8   0:00.15 vim
 9802 user        20   0   49172   8576  5668  S   0.0   0.8   0:00.15 vim
  
```

(c) Patricio Domingues

- Ficheiro `/proc/stat`
 - Mantém estatísticas sobre atividade do sistema
 - Utilitário **watch** para execução periódica de linhas de comando
 - `-n 2`: todos os 2 segundos
- ```
watch -n 2 cat /proc/stat
```

[illegible]

# Por falar em Linux...

✓ 25 de agosto de 1991...

– Fonte: <http://www.linux-netbook.com/linux/timeline/>

## Interactive Timeline of the History of Linux



August 25, 1991

### **I'm doing a (free) operating system**

The 21 year old Finnish student Linus Benedict Torvalds **announced his work** on a free operating system in the comp.os.minix Usenet newsgroup.

Linus' own assessment, of where this would go in the future, could have hardly been further off:

*I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones.*

SEPTEMBER  
1, 1991  
linux 0.0.1 released

Linus Torvalds Tribute by Noel Merino



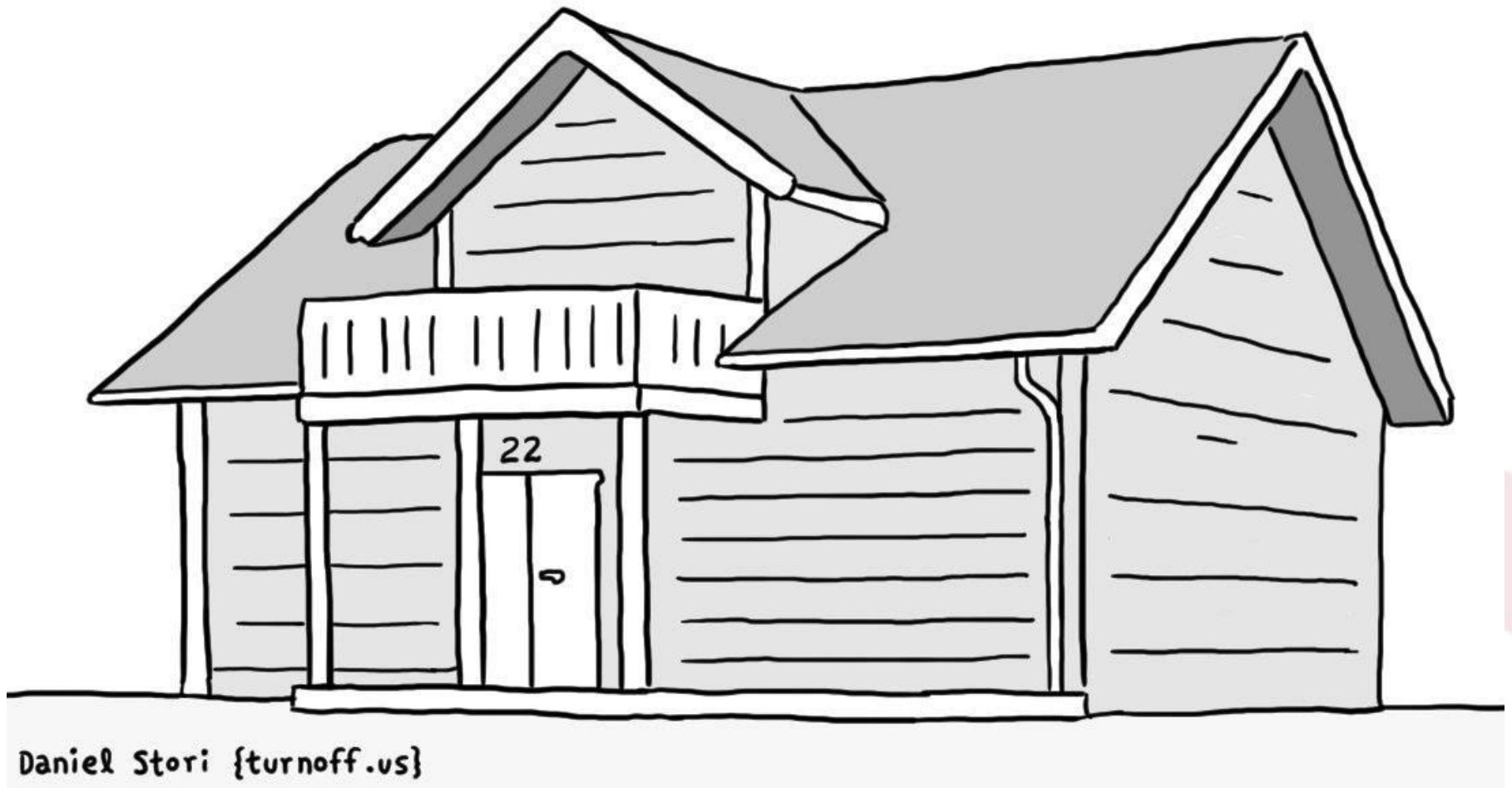




**IPL**

escola superior de tecnologia e gestão  
instituto politécnico de leiria

Meanwhile, at Linus Torvald's house...



Daniel Stori {turnoff.us}

# Plataforma Linux

- 20 e tal anos depois...
- Linux está em muitas áreas da informática...
- Sistemas servidores
  - DNS, DHCP, SMTP, IMAP, HTTP, Proxy, etc.
- Equipamentos informáticos
  - Routers domésticos, sistemas web cams, domótica,...



- Sistemas entretenimento
  - Indústria automóvel, aviação, etc.
- Veículos autónomos
- Supercomputadores
- Computação móvel (Android)
- “Smart” TVs





## • Kernel

- Conjunto de software que é responsável pela interação com a máquina
  - Interage com hardware
  - Providencia serviços ao software
- O kernel é controlado pelo Linus Torvalds
- Código fonte disponível (kernel.org)
  - Formato .xz
    - similar ao 7z mas com metadados adaptados a sistemas Unix
  - [www.kernel.org](http://www.kernel.org)

## • Antigamente...

- A numeração do kernel indicava se era *estável* ou de *desenvolvimento*
  - 2º número de versão é par
    - 2.4: estável
    - 2.3: desenvolvimento

## • Esquema abandonado

## • Novo esquema

|             |                |            |
|-------------|----------------|------------|
| mainline:   | 5.0-rc7        | 2019-02-18 |
| stable:     | 4.20.10        | 2019-02-15 |
| longterm:   | 4.19.23        | 2019-02-15 |
| longterm:   | 4.14.101       | 2019-02-15 |
| longterm:   | 4.9.158        | 2019-02-15 |
| longterm:   | 4.4.174        | 2019-02-08 |
| longterm:   | 3.18.134 [EOL] | 2019-02-06 |
| longterm:   | 3.16.63        | 2019-02-11 |
| linux-next: | next-20190219  | 2019-02-19 |

Latest Stable Kernel:  
↓ 4.20.10





## Linux Cross Reference

Free Electrons  
Embedded Linux Experts

• Source Navigation • Identifier Search • Freetext Search •

Version: 2.0.40 2.2.26 2.4.37 3.8 3.9 3.10 3.11 3.12 3.13 3.14 3.15 3.16 3.17 3.18 3.19 4.0 4.1 4.2 4.3 **4.4**

Linux/

- Documentation/
- arch/
- block/
- certs/
- crypto/
- drivers/
- firmware/
- fs/
- include/
- init/
- ipc/
- kernel/
- lib/
- mm/
- net/
- samples/
- scripts/

- Kernel
  - Navegação pelo código fonte
  - <http://lxr.free-electrons.com/>
- A seguir:
  - linux kernel map



## • Distribuição

- Conjunto de software que forma um sistema operativo

- Kernel
- Aplicações do sistema (arranque, init ou systemd, etc.)
- Aplicações para interação com o utilizador
  - Shell, windows manager
- Aplicações para o utilizador
- Gestor de aplicações
- ...

- Existem centenas de distribuições de linux

- Mais conhecidas

- Ubuntu, debian, red hat, suse, slackware, gentoo, mint...



- Algumas específicas

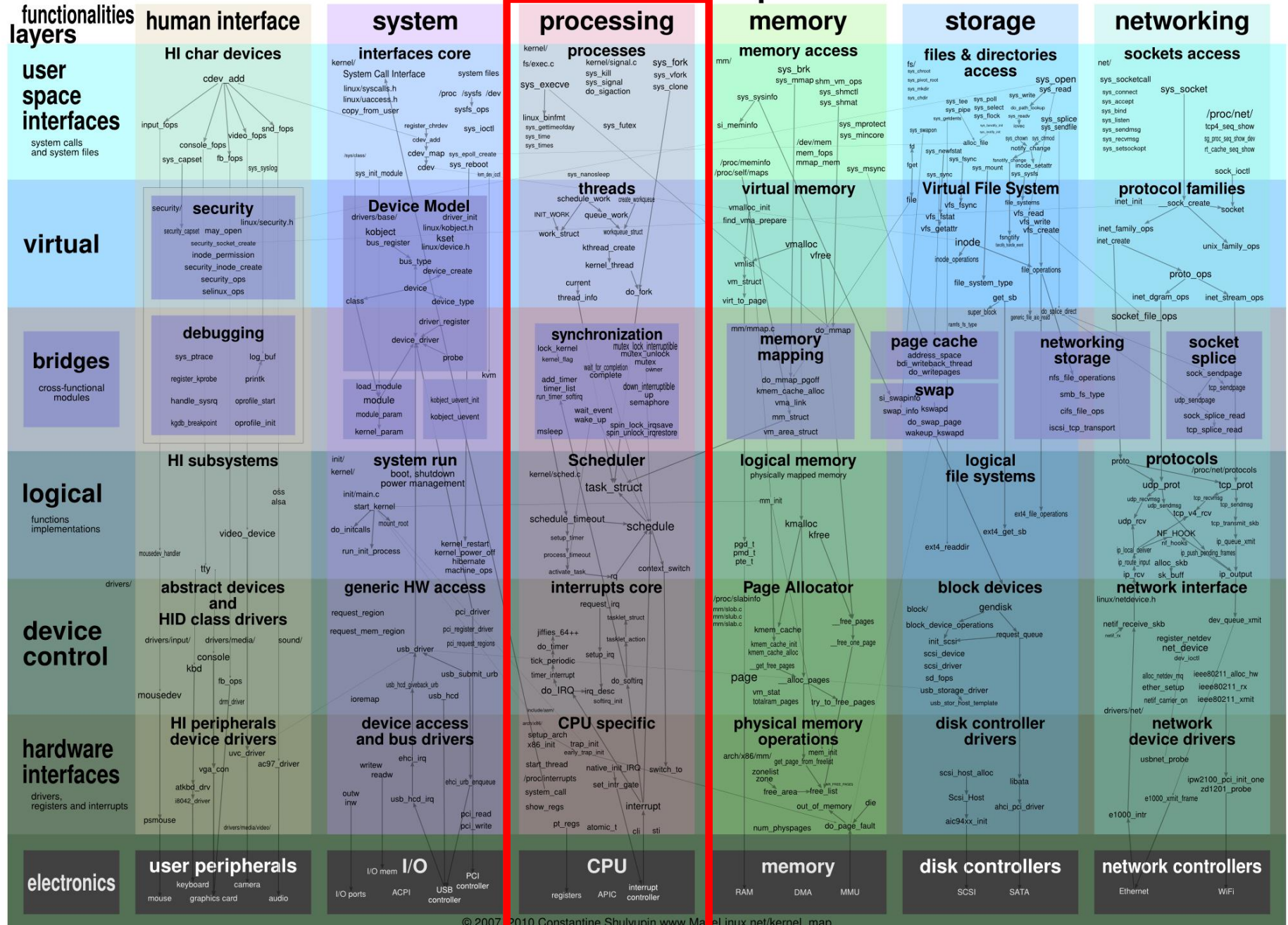
- Kali, CAINE (forensics), DEFT (Digital Evidence & Forensic Toolkit), openWRT, damn small linux, puppy linux,...
- <https://distrowatch.com/>

**IPL**

escola superior de tecnologia e gestão  
instituto politécnico de leiria

# [http://www.makelinux.net/kernel\\_map/](http://www.makelinux.net/kernel_map/)

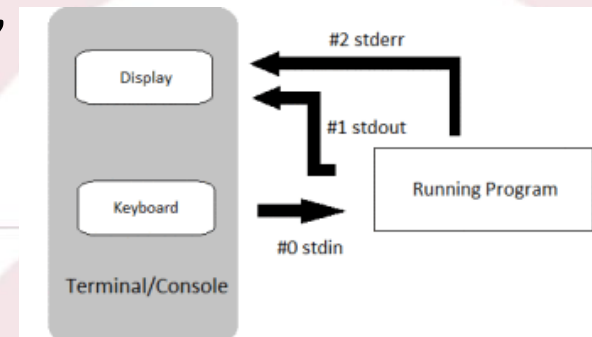
## Linux kernel map



# Elementos de um processo (1)

- ✓ Existem dois elementos essenciais num processo
  - Código do programa
    - Pode ser partilhado com outros processos que executam o mesmo programa
  - Um conjunto de dados associados ao processo
- ✓ Exemplo
  - Quatro processos a executar o *notepad*
    - Partilham o código binário
    - Cada processo tem os seus dados
      - conteúdo do ficheiro em edição no notepad, posição do cursor, etc.

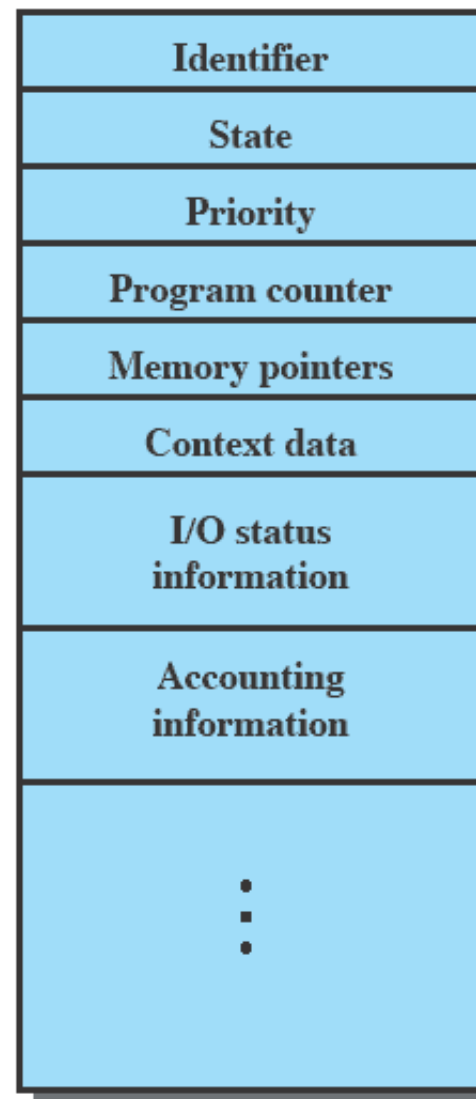
- ✓ Alguns elementos caracterizadores de um processo
  - Identificador (ProcessID, PID)
  - Prioridade
  - Estado do processo
  - Registos da CPU
  - Informação sobre operações de Entrada/Saída
  - Contador de programa (“Program Counter)
  - Contabilização de recursos
    - Tempo de CPU consumido pelo processo,
  - Tabela de ficheiros abertos





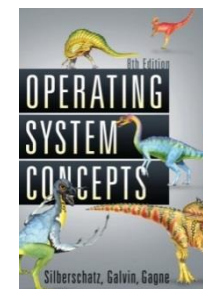
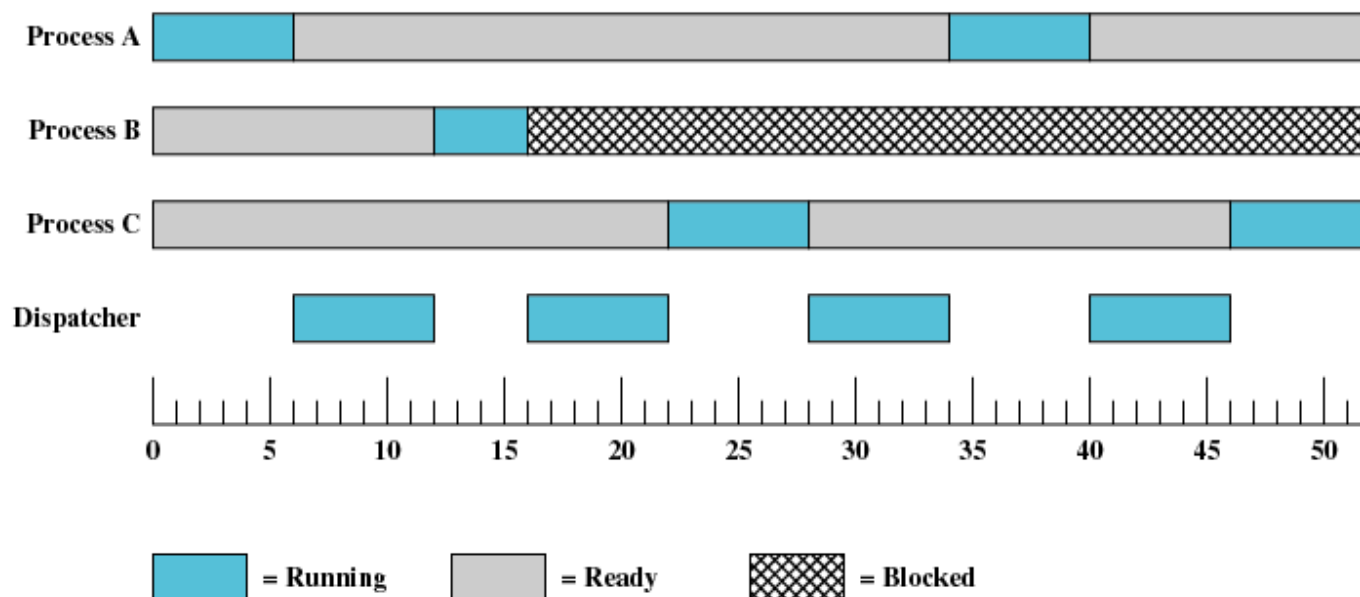
# Process Control Block - PCB

- ✓ Process Control Block – PCB
  - Designa os elementos do processo
  - PCB é criado e mantido pelo SO
  - PCB é essencial para sistemas multitarefas
    - Permite a interrupção da execução do processo e o posterior restaurar da execução do mesmo



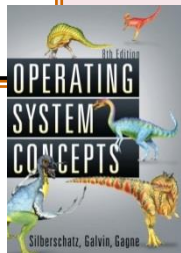
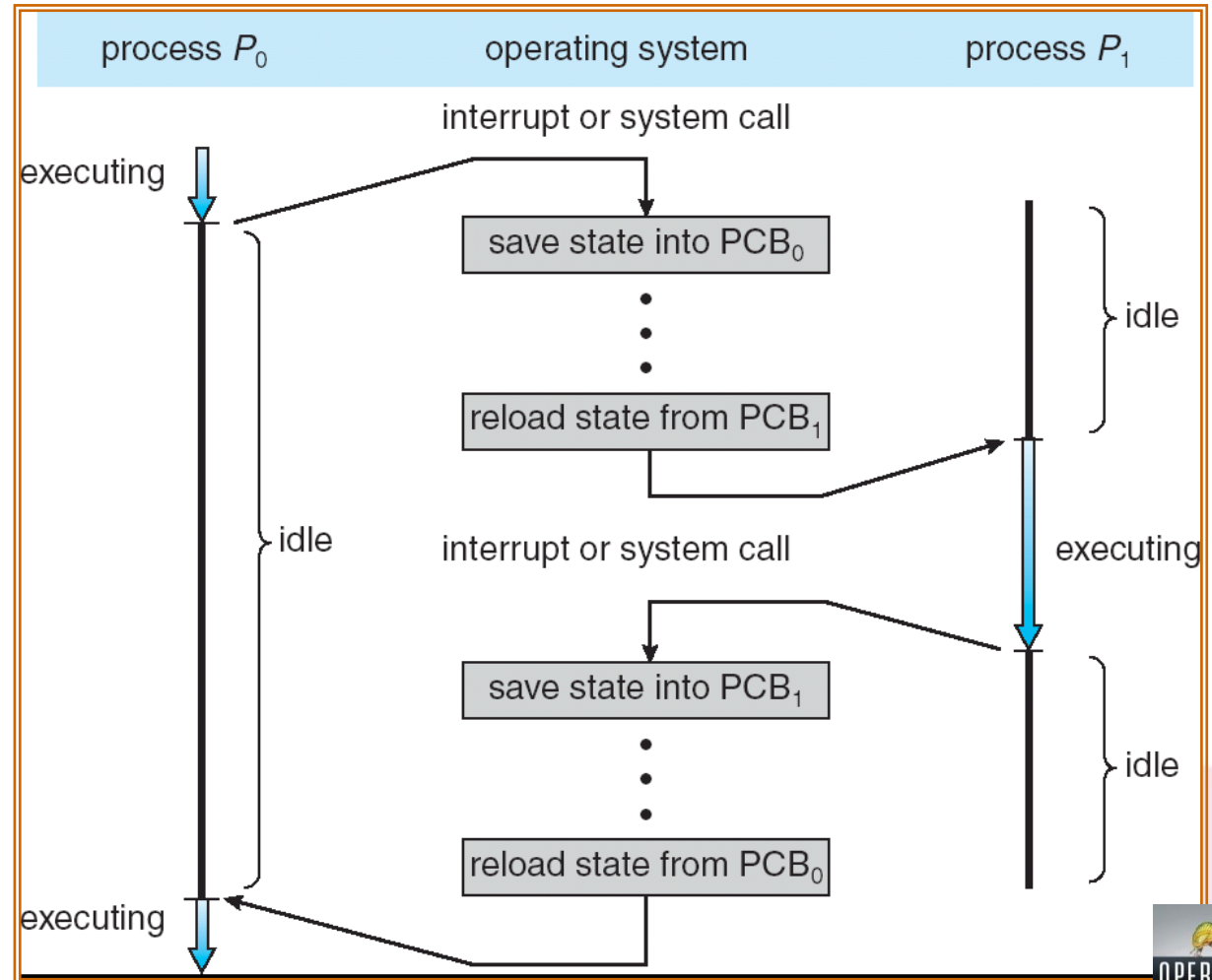
# Comutação de processos

- ✓ Um SO multitarefa vai comutando os processos pelo CPU
  - Em cada instante, num single core, apenas é executado um processo
  - Num determinado intervalo de tempo (e.g., 1 segundo) passam vários processos (múltiplas vezes) pelo CPU
    - Utilizador com a ilusão de vários processos em execução simultânea
    - Está ao mesmo tempo a compilar um programa, escrever no word, a efetuar um *download*, a ouvir um MP3, anti-vírus em execução, etc.
  - A comutação é da responsabilidade do escalonador do SO



# Operação de comutação

- ✓ Comutação do processo “P0” para o processo “P1”
  - “P0” passa de execução para “ready”
  - “P1” passa de “ready” para execução

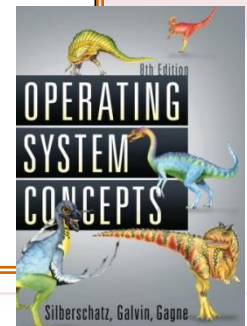
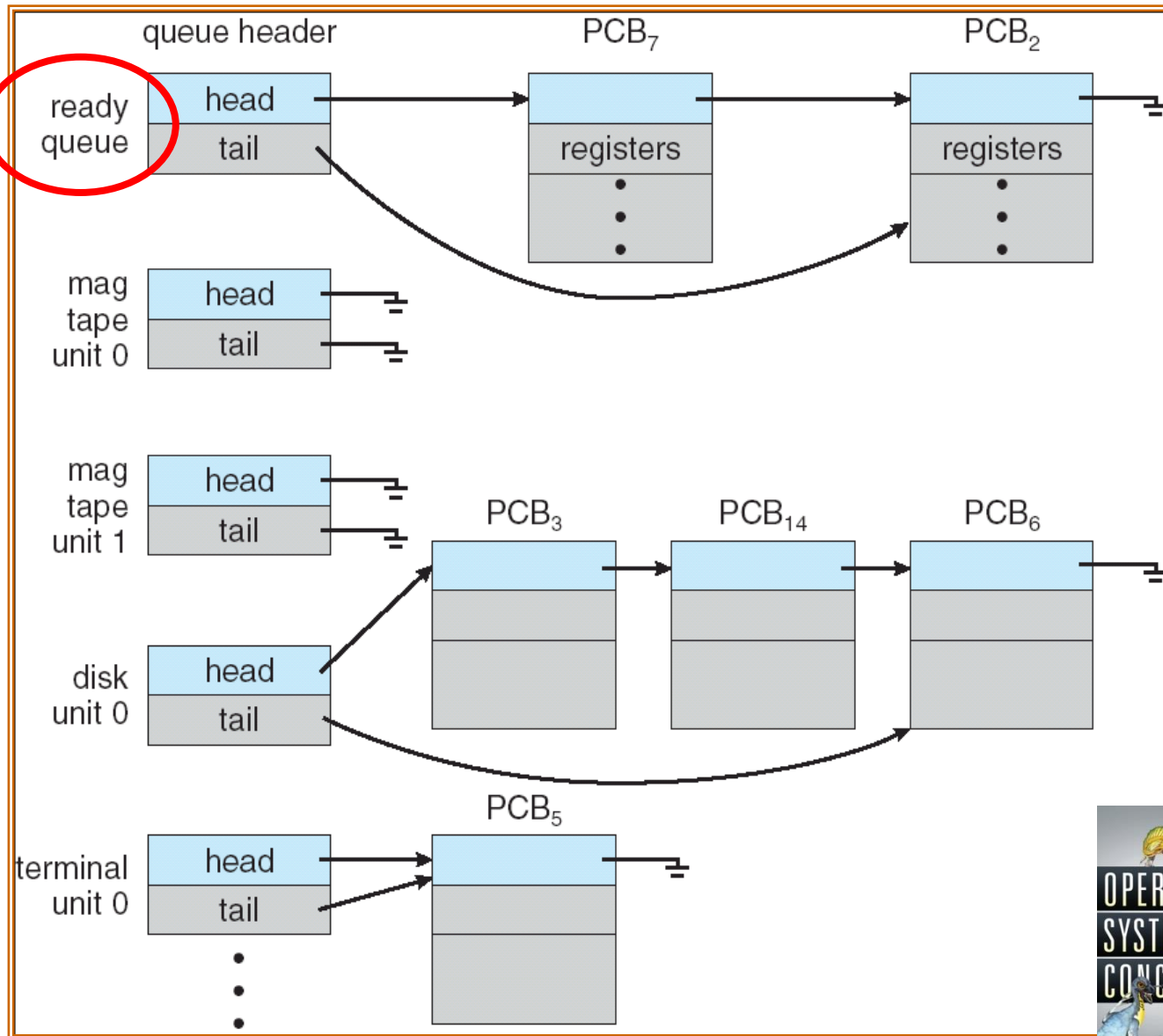


# Filas de processos

- ✓ Os PCBs são mantidos em várias filas de esperas (“queues”)
- ✓ Consoante o estado do processo, os PCBs migram de uma fila para a outra
- ✓ **Processos “prontos”**
  - Conjunto dos processos que estão “prontos” para serem executados
    - Aguardam apenas pelo CPU
- ✓ **Processos à espera de E/S**
  - Conjunto de processos que aguardam dispositivos de E/S
    - E.g., término da leitura de um bloco em disco, de um pacote de rede
  - Também conhecidos como processos “bloqueados”

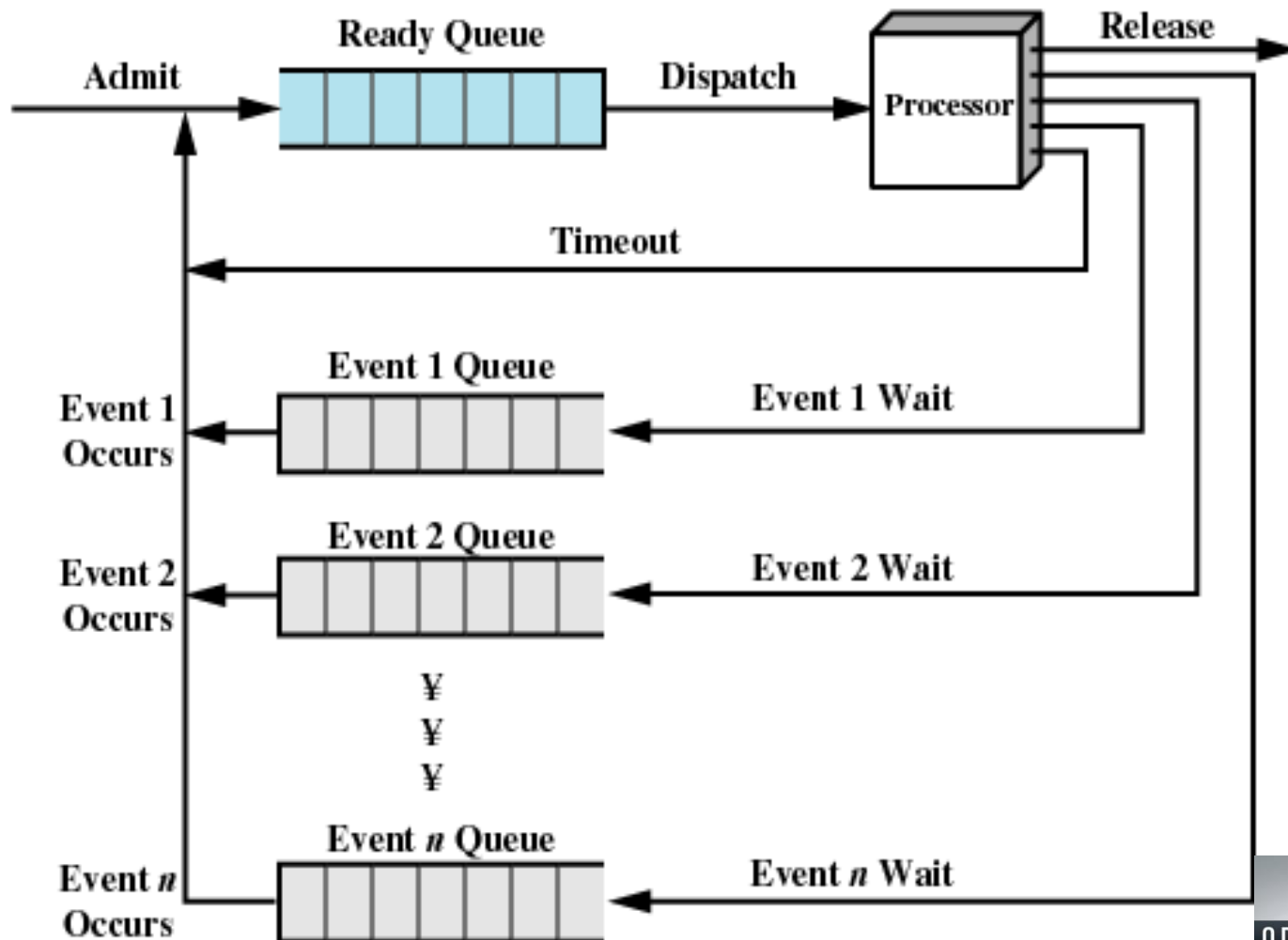
# As várias filas de processos

Fila dos processos “bloqueados”  
(fila organizada por dispositivo de E/S)



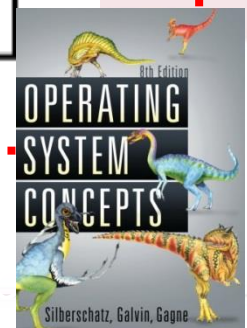


# Os PCBs vão comutando de filas...



**Escalonador de processos**

How that events do not have  
and to / ... (more)



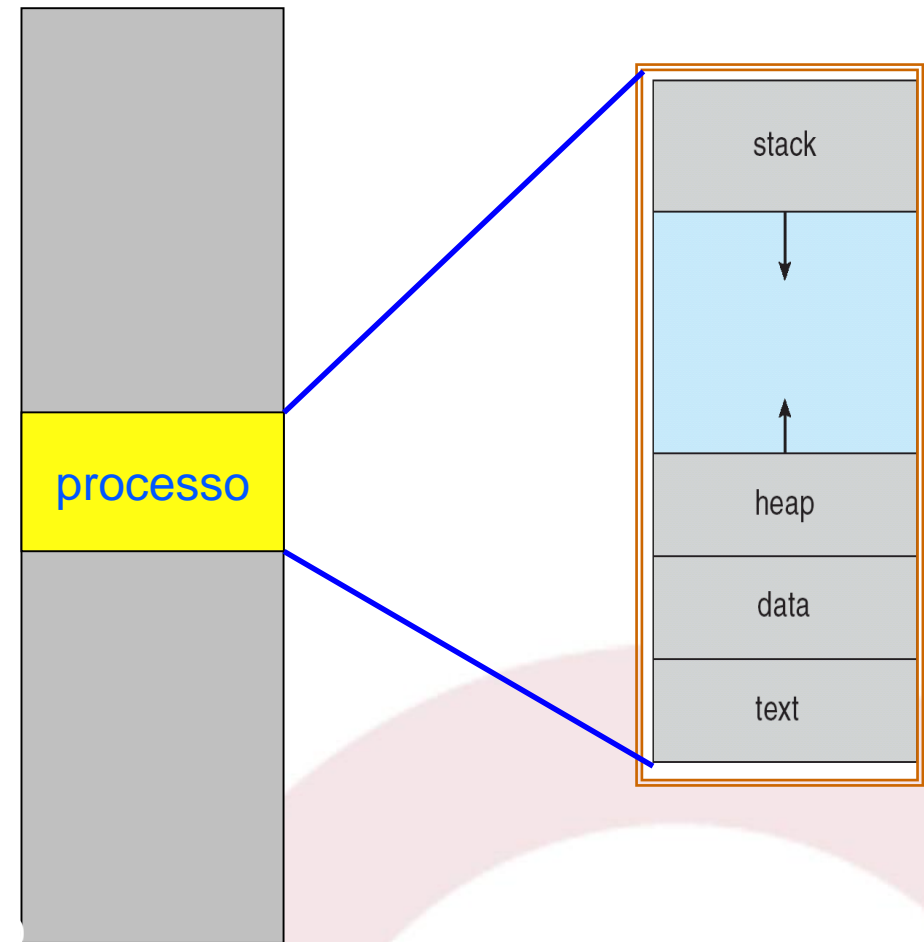
- ✓ O SO mantém tabelas sobre as várias entidades que controla
  - Memória
  - Dispositivos
    - Tabelas de E/S
  - Ficheiros
    - Tabelas de ficheiros abertos
  - Processos
    - Tabela de processos
      - Cada elemento aponta para a imagem do processo

Slide seguinte: Imagem de um processo >>



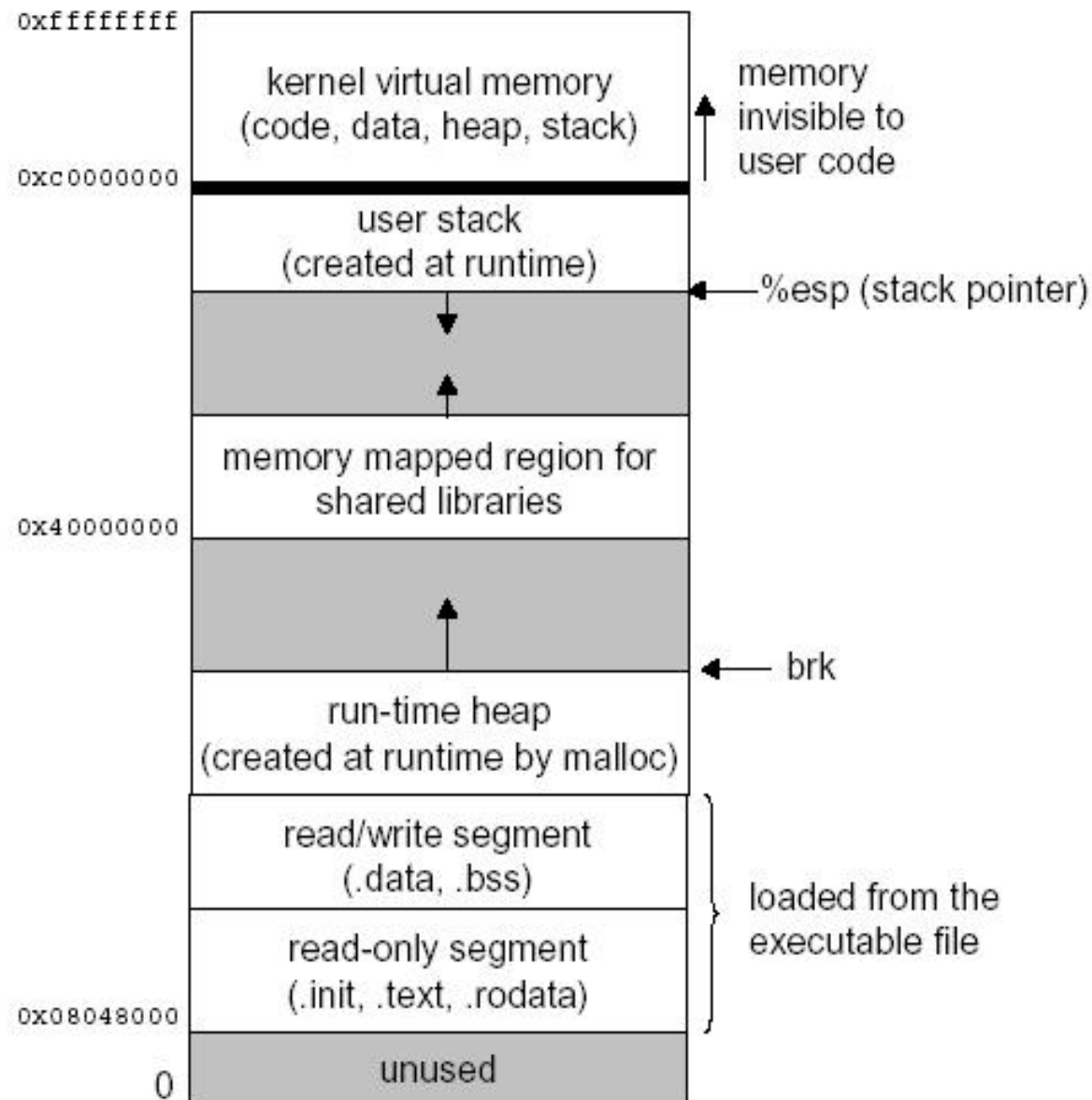
- ✓ A imagem de um processo em memória é composta por vários segmentos
  - Segmento de texto
  - Segmento de dados
  - Segmento “heap”
  - Segmento “stack”
- ✓ Memória virtual

Espaço endereçamento



# Imagem do processo (2)

- ✓ Imagem de um processo em Linux
  - Espaço endereçamento de 32 bits
  - Fonte:  
<http://tinyurl.com/2mwd>



- ✓ Imagem do processo
  - Representação do processo na memória
  - Divide-se em vários segmentos
- ✓ Segmento “*text*”
  - Contém o código do processos (instruções que são executadas)
  - Identificada com a secção “.text”
- ✓ Segmento “*data*”
  - Contém variáveis, dividindo-se em duas secções
  - .data: variáveis globais e estáticas (“static”) iniciadas valor diferente de zero
  - .bss: variáveis globais e estáticas não iniciadas (valor zero)



## ✓ Segmento “*heap*”

- Zona da memória dinâmica
  - De onde provém a memória alocada (exemplo: `malloc()` )
  - Segmento de tamanho variável

## ✓ Segmento “*stack*”

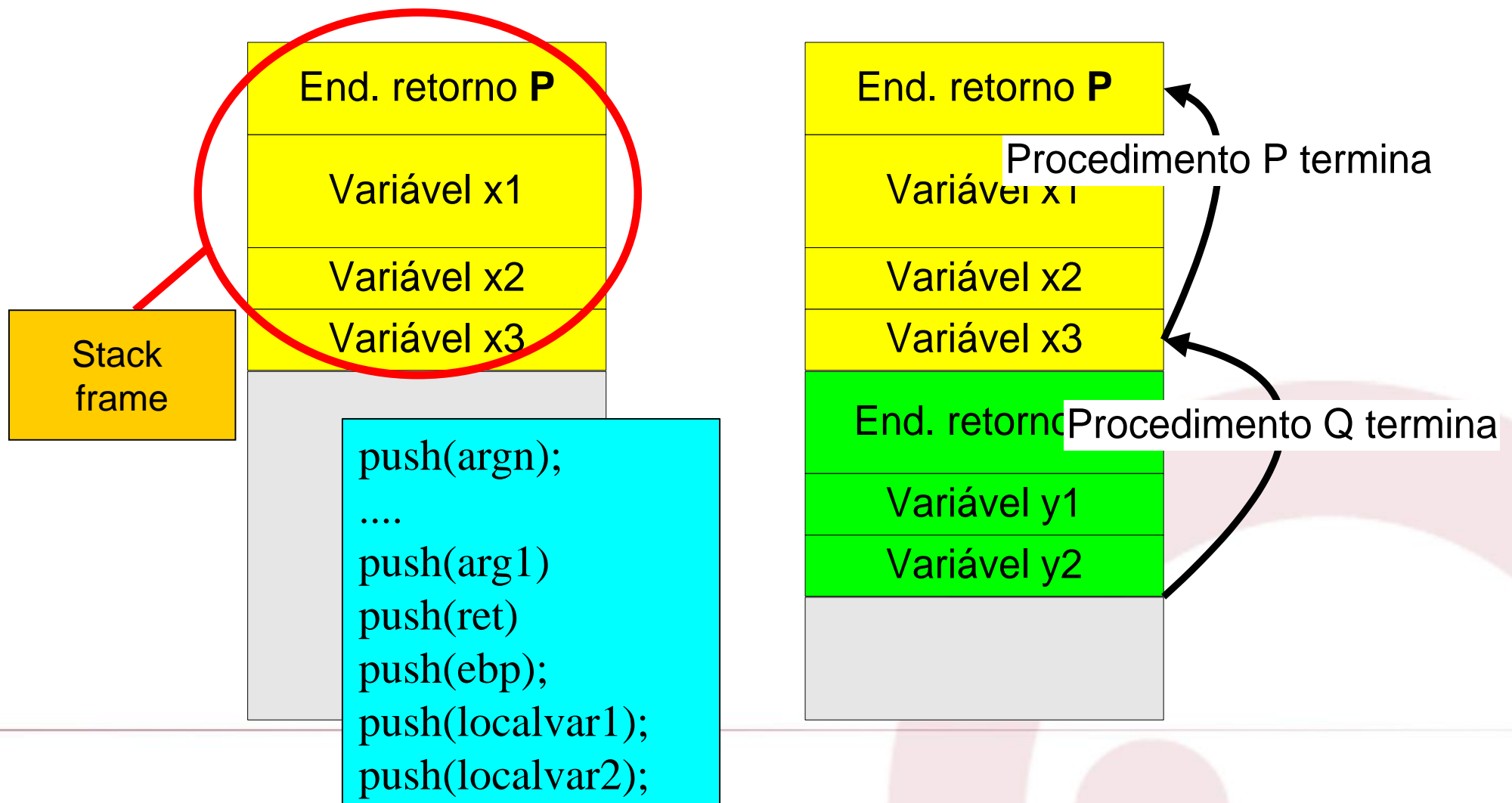
- Zona das variáveis “automáticas” (ou “locais”)
- Variáveis que são automaticamente criadas e destruídas nas funções e métodos
- Segmento de tamanho variável



- ✓ O segmento de pilha serve para...
  - Passagem de parâmetros em procedimentos e funções
  - Guardar o endereço de retorno de procedimentos e funções
  - Guardar as variáveis ditas automáticas
    - As variáveis locais de um procedimento ou função são:
      - criadas no segmento de pilha no início do procedimento ou função
      - destruídas no fim do procedimento ou função

# Utilidade do segmento de *stack* (2)

- ✓ Exemplo - chamada de dois procedimentos
  - Procedimento P chama o procedimento Q



# Particularidades da *stack* (1)

- ✓ Quando é chamada uma função, é acrescentado um *stack frame* novo à *stack* do processo
- ✓ Quando termina a função, o respetivo *stack frame* é terminado
  - Isso implica que...
    - As variáveis locais de uma função deixam de existir depois do término de uma função
    - É **errado** retornar o endereço de uma variável local
      - Quando a função terminar, o endereço continua a existir, mas a variável já não...

- ✓ O uso de *stack* permite recursividade
  - Função que se chama a si próprio
  - Útil para a resolução de certos problemas
    - Exemplo: percorrer uma árvore binária, etc.
- ✓ Recursividade
  - Cada nova chamada origina um novo *stack frame*



Código assembler da função  
(objdump -d ficheiro.exe)

## ✓ Código recursivo\_1.c

```
#include <stdio.h>
void recursivo_non_stop(int x){
 printf("x=%d\n", x);
 recursivo_non_stop(x+1);
}
int main(void){
 recursivo_non_stop(0);
 return 0;
}
```

```
0804841d <recursivo_non_stop>:
804841d: push %ebp
804841e: mov %esp,%ebp
8048420: sub $0x18,%esp
8048423: mov 0x8(%ebp),%eax
8048426: mov %eax,0x4(%esp)
804842a: movl $0x8048500, (%esp)
8048431: call 80482f0 <printf@plt>
8048436: mov 0x8(%ebp),%eax
8048439: add $0x1,%eax
804843c: mov %eax, (%esp)
804843f: call 804841d <recursivo_non_stop>
8048444: leave
8048445: ret
```

## ✓ Execução: 261849 chamadas antes do transbordo de pilha (*stack overflow*)

x=261849

x=261850

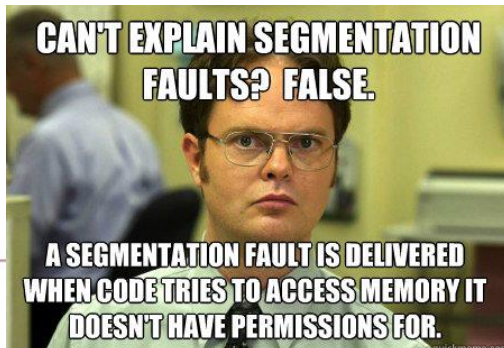
Segmentation fault (core dumped)

Segmentation fault >>



# O que é um *segmentation fault*?

- Detecção de tentativa de acesso de um processo a um espaço de memória que não lhe pertence
  - Causas: ponteiro perdido, uso de memória anteriormente libertada (free), transbordo de memória, etc.
- Hardware (unidade de gestão de memória) deteta falha e notifica o SO
- SO atua
  - Unix: envio do signal `SEGFAULT` ao processo
  - Por omissão, o signal termina o processo
    - Exceto se o processo estiver a capturar o signal
- É ainda produzido um ficheiro “core dump”
  - Contém o conteúdo da memória do processo para depuração
- Máquina virtual SO
  - Escrita core para disco requer que a opção `-c` do `ulimit` esteja ativa



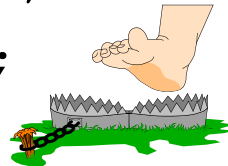
- `ulimit -c unlimited`

# Código C mal comportado...

```
int main(void) {
 int *RetPtr;
 RetPtr = RetEnderecoVariavelLocal(10);
 printf("#1 - %d\n", (*RetPtr));
 /* Chamada recursiva */
 Recursivo();
}

int *RetEnderecoVariavelLocal(int x1) {
 int Total;
 Total = x1 * 1.235;
 return (&Total);
}

void Recursivo(void) {
 /* chamada recursiva */
 Recursivo();
}
```



# Mapeamento “código fonte” >> imagem

- ✓ Código fonte C com a indicação para cada identificador / variável do segmento que ocupa na imagem do processo na memória

```
#define KB (1024) /* préprocessador - não existe na imagem do processo */
#define MB (1024*1024)
char buf[10*MB]; /* global, não inicializado → .bss */
char command[KB] = "command?"; /* global, inicializado → .data */
int n_lines = 0; /* global, inicializado a zero → .bss */
int n_tries = 20; /* global, inicializado → .data */
int f(int n) {
 int result; /* local → stack */
 static int number_calls = 0; /* static, inicializado a zero → .bss */
 ++number_calls; result = n*n;
 return result;
}
int main() {
 int x = 5; /* local → .stack */
 printf("f(%d)=%d\n", x, f(x));
 return 0;
}
```

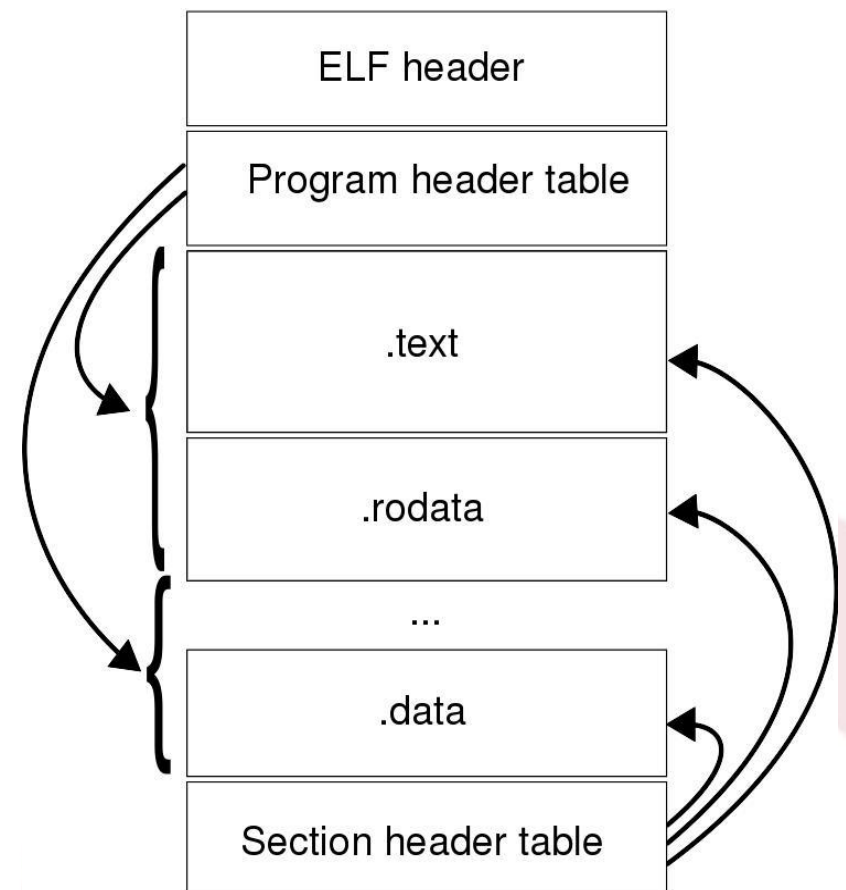
# E o ficheiro executável? (1)

- ✓ Um ficheiro executável (.exe) contém os dados necessários à criação do processo que executa uma instância da aplicação
- ✓ Um executável contém todos os segmentos, exceto o **.bss**
  - O .bss é representado de forma compacta no ficheiro executável
    - .bss contém as variáveis globais / estáticas não inicializadas
    - Apenas é guardado o tipo da variável e o seu tamanho, não sendo guardado o seu conteúdo pois não está inicializado
    - Poupa-se espaço no ficheiro executável
    - Exemplo: `int A[100000];`
      - não ocupará espaço no executável ( $A = 100000 * \text{sizeof}(A)$ )

# Ficheiro executável: formato ELF (1)



- ELF: Executable and Linkable Format
- Formato de ficheiro empregue no Linux/Mac OS X
  - Ficheiros executáveis
  - Bibliotecas dinâmicas
  - Ficheiros de código objeto
  - *Core dumps*
- O formato ELF é dual
  - Compiladores e “linkers” tratam como um conjunto de secções lógicas descritas por uma tabela de secções
  - O executor de programas do SO vê o ficheiro como um conjunto de segmentos descritos por uma tabela de programa
- `.rodata`
  - “read only data”

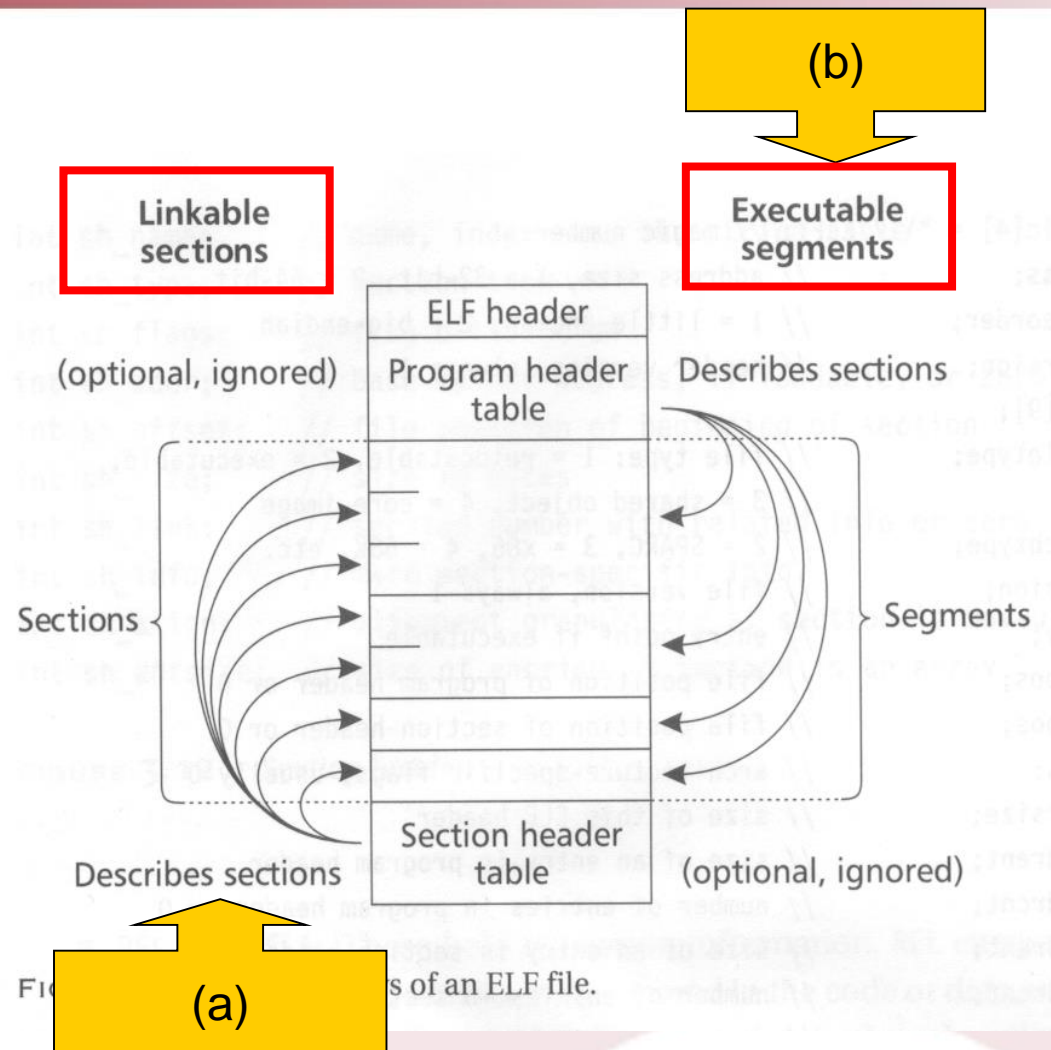


Continua no próximo slide >>



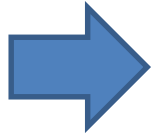
# Ficheiro executável: formato ELF (2)

- O formato ELF é dual
- (a) Compiladores e “linkers” tratam como um conjunto de secções lógicas descritas por uma tabela de secções
- (b) O executor de programas SO vê o ficheiro como um conjunto de segmentos descritos por uma tabela de programa

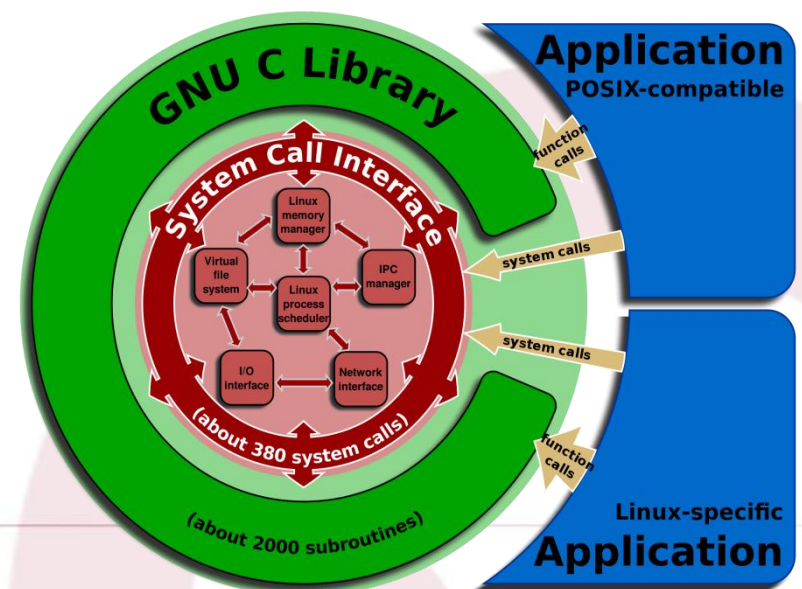


# glibc - Biblioteca do C

- As funções do C (printf, strcpy, etc.) existem na “biblioteca do C”
- Em sistemas GNU (e.g., Linux)
  - glibc
- Outras “libc”
  - Microsoft C Run-time Library
  - µClibc, µClinux, newlib para sistemas embebidos
  - Bionic (android)
  - ...



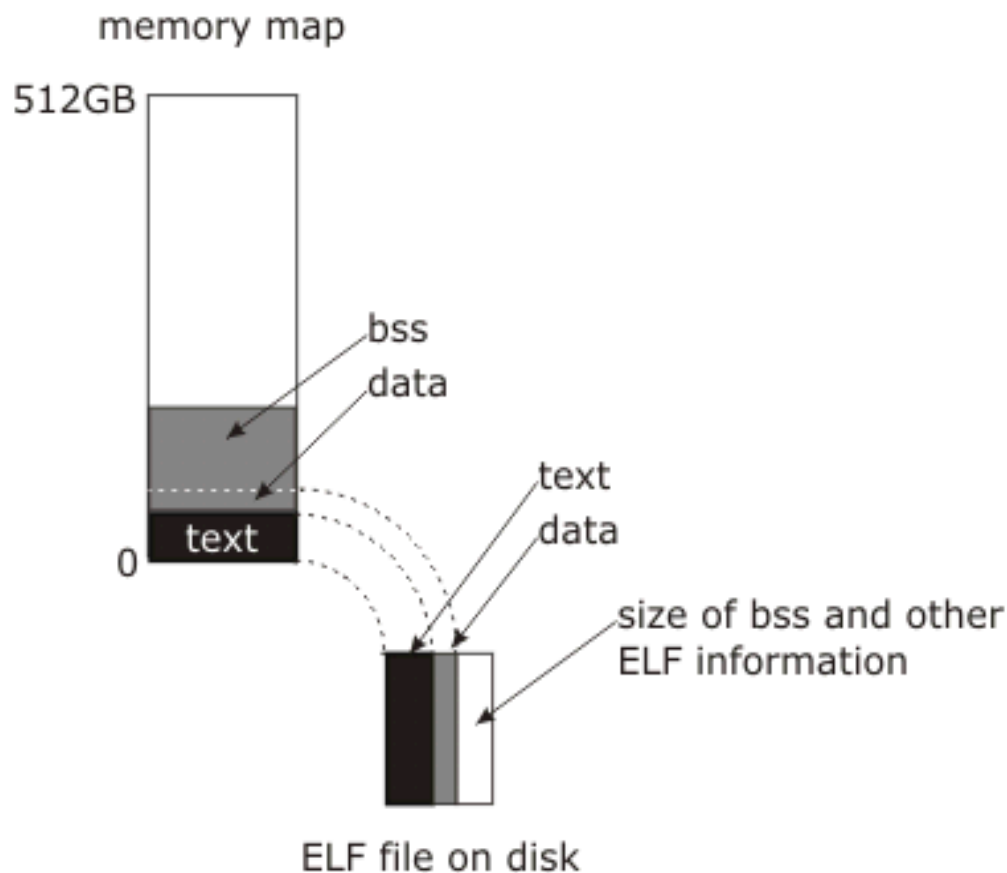
```
user@ubuntu: /
File Edit Tabs Help
user@ubuntu:/lib/i386-linux-gnu$ ls -l libc-2.19.so
-rwxr-xr-x 1 root root 1758972 Apr 12 2014 libc-2.19.so
```



# E o ficheiro executável?

## ✓ BSS: Block Started by Symbol

- Segmento de dados que contém as variáveis “static” e “globais” iniciadas a zero
- Não ocupa espaço no ficheiro executável (EXE)



## ✓ Segmento “data”

- Variáveis “static” e “globais” iniciadas com valor diferente de zero
- Existe o segmento “data” no ficheiro executável

# Estados possíveis de um processo

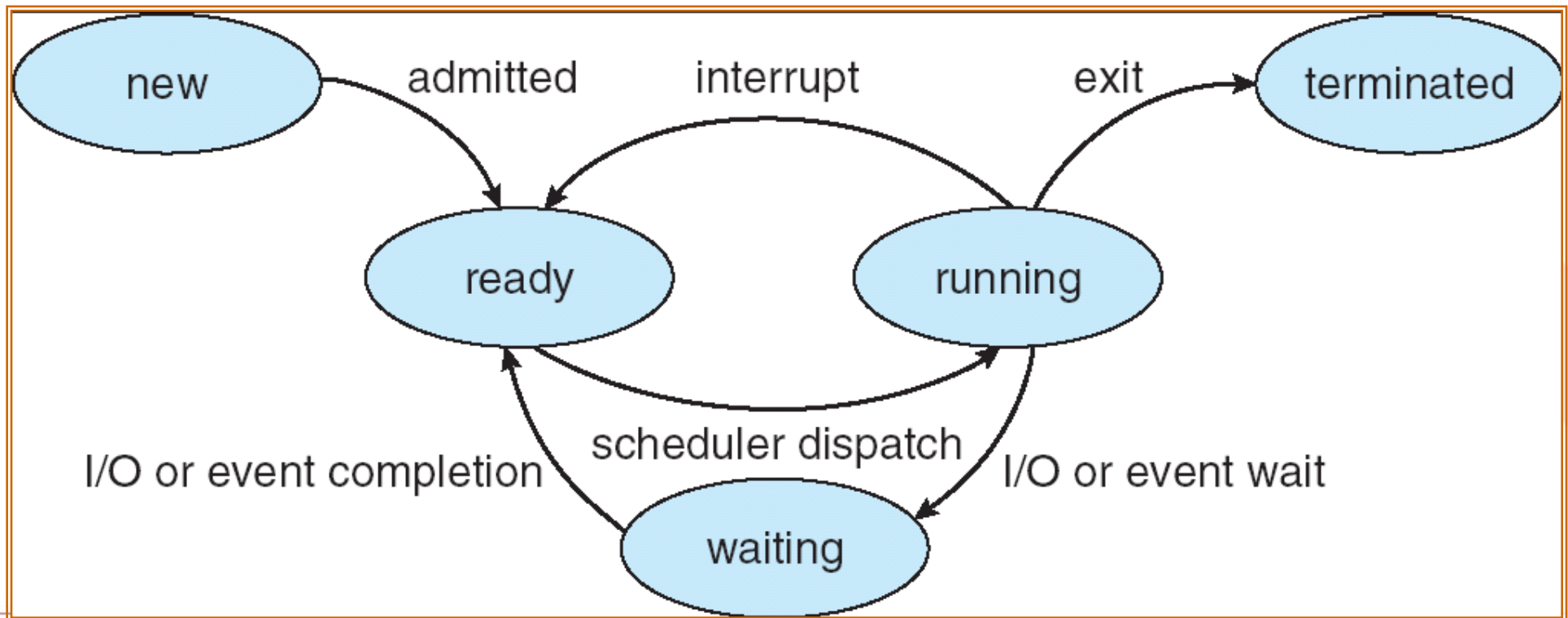
- ✓ Os estados de um processo podem ser representados por uma máquina de estados finito
  - Máquina de estados finitos
    - Conjunto de estados e eventos
    - Eventos desencadeiam a mudança de estados
  - Máquina de estado de um processo
    - Estados possíveis de um processo
    - Eventos que levam o processo a mudar de um estado para o outro

**Máquina de estados de um processo (exemplo simplificado)>>**

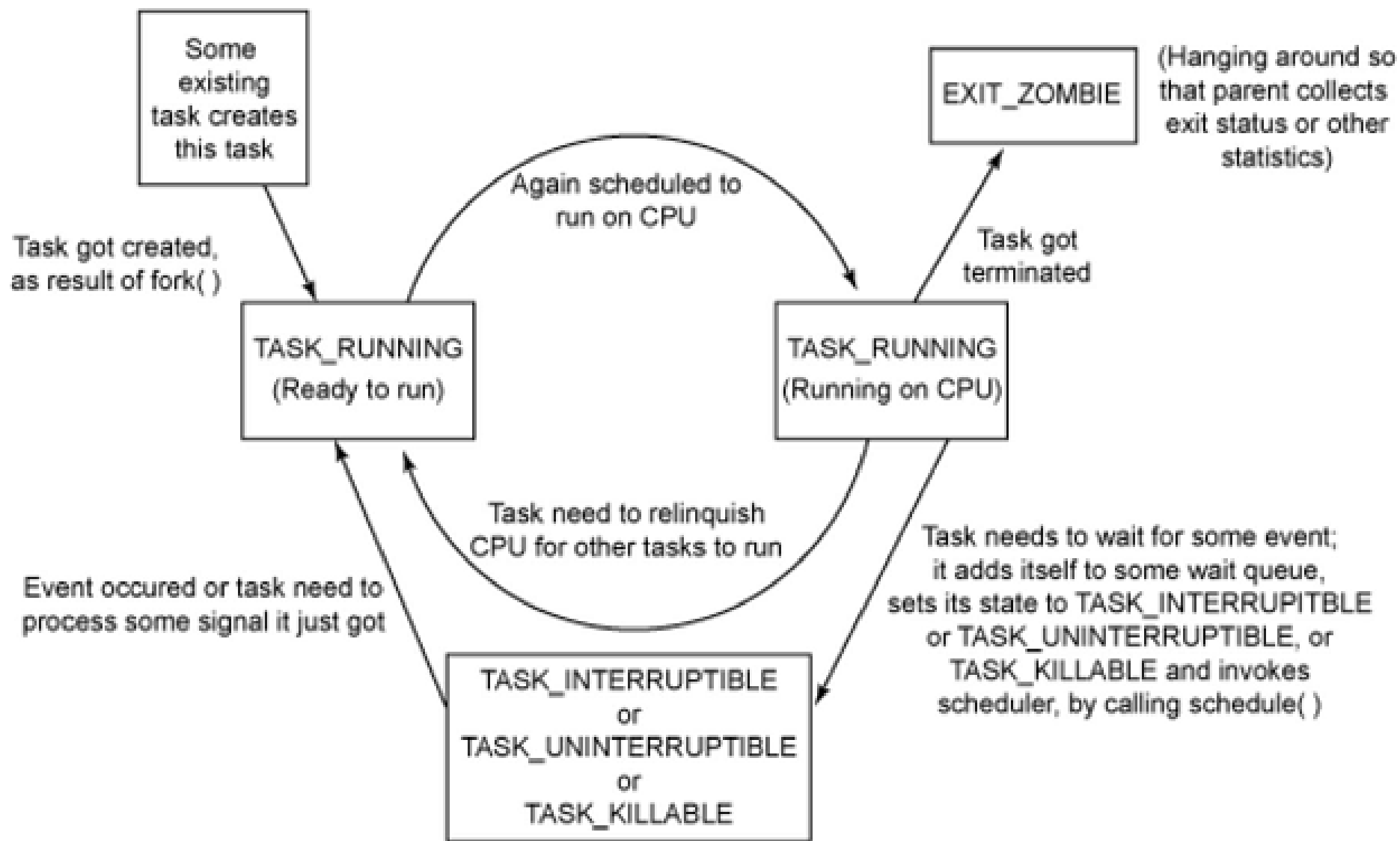


# Máquina de estados de um processo

- Diagrama (simplificado) de uma máquina de estados de um processo
- O processo inicia no estado “new” e depois transita por vários estados (consoante os eventos) até terminar (“terminated”)
- Um processo “ready” tem tudo para executar, exceto o processador



# E no Linux? (processo $\approx$ task)



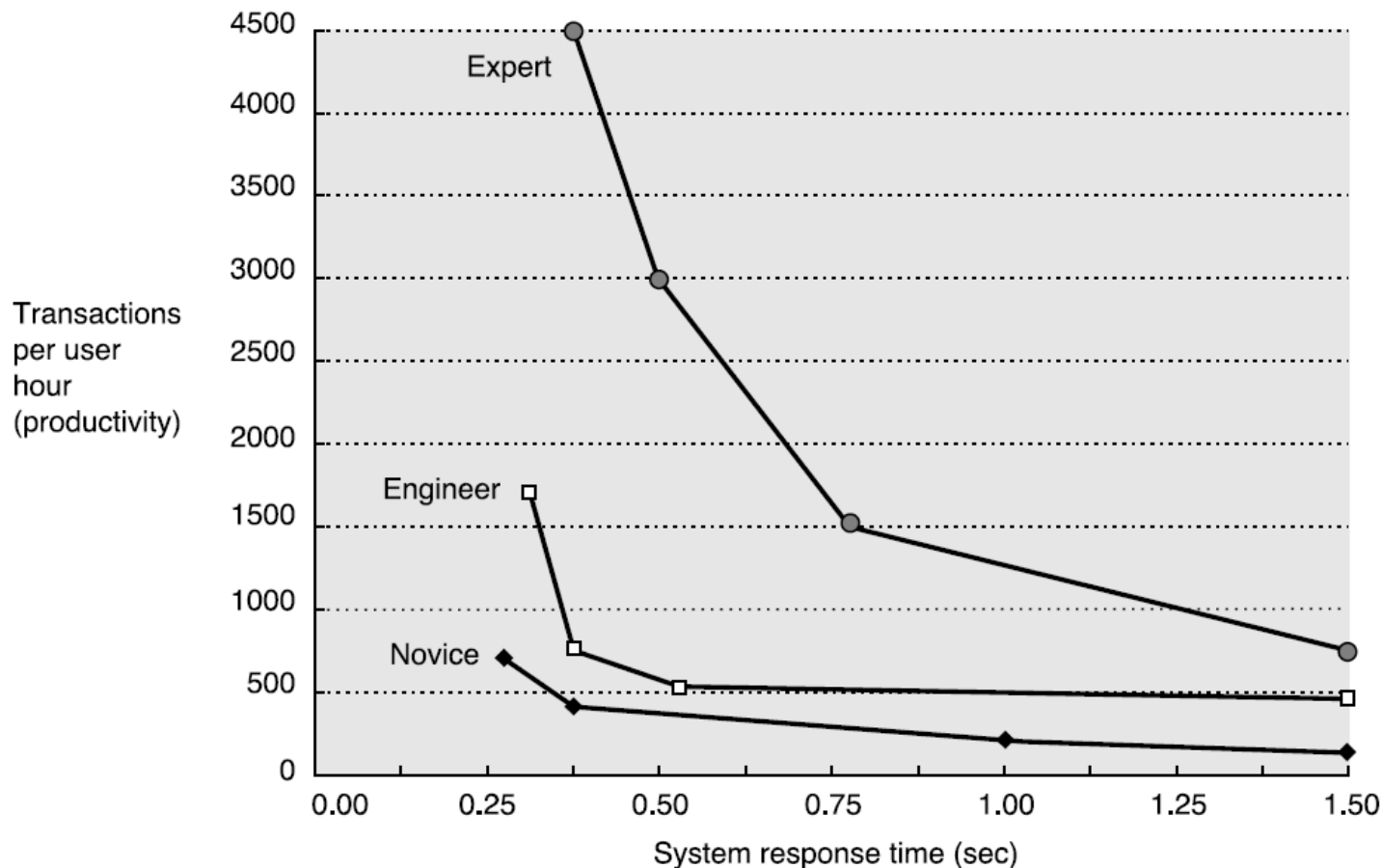


- ✓ A que se deve uma abordagem orientada aos eventos?
  - O CPU é muito mais rápido do que os componentes de E/S

| Característica    | Escala computador | Escala humana           |
|-------------------|-------------------|-------------------------|
| CPU               | 2 GHz             | Não considerado         |
| Ciclo de relógio  | 0.5 nanosegundos  | 1 s                     |
| Acesso à cache L2 | 10 nanosegundos   | 20 s                    |
| Acesso à memória  | 80 nanosegundos   | 160 s (2,6 minutos)     |
| Acesso a disco    | 8 milissegundos   | 16000000s (185 dias)    |
| Quantum processo  | 100 milissegundos | 200 000 000s (6.3 anos) |

# Sistemas interativos

- SO multitarefas devem ter cuidado no nível de multitarefa a suportar pelo sistema
  - O tempo de resposta é muito importante para o utilizador
    - Tempo de resposta elevados levam utilizador à impaciência





# “Man shoots his computer”...

✓ O que pode suceder quando o computador não quer colaborar...

– <http://www.bbc.com/news/world-us-canada-32407688>

## Enraged US man shoots his malfunctioning computer

A man in the US city of Colorado Springs faces police action after becoming so frustrated with his computer that he took it outside and shot it eight times, police say.

"He was having technology problems, so he took it to the back alley and destroyed it," a police spokesman said.

Lucas Hinch was briefly detained for discharging a firearm within the city.

He did not realise he was breaking the law when he went "Wild West" on his machine, local media reported.

A judge is due to decide what penalty he will receive.

"He got tired of fighting with his computer for the last several months," police spokesman Jeff Strossner told the **Colorado Springs Gazette**.

The paper said that Mr Hinch "shot the darn thing" when ctrl+alt+delete - the traditional method used to re-boot computers - "consistently did not work" on Monday evening.

"He was able to wreak the kind of revenge most of us only dream about," the paper said. "The computer is not expected to recover."



- ✓ Classificação consoante os recursos usados pelo processo
  - Processo dependente de E/S (“I/O bound”)
  - Processo dependente de CPU (“CPU bound”)

## Próximos slides

Processo dependente de E/S e processo dependente de CPU >>



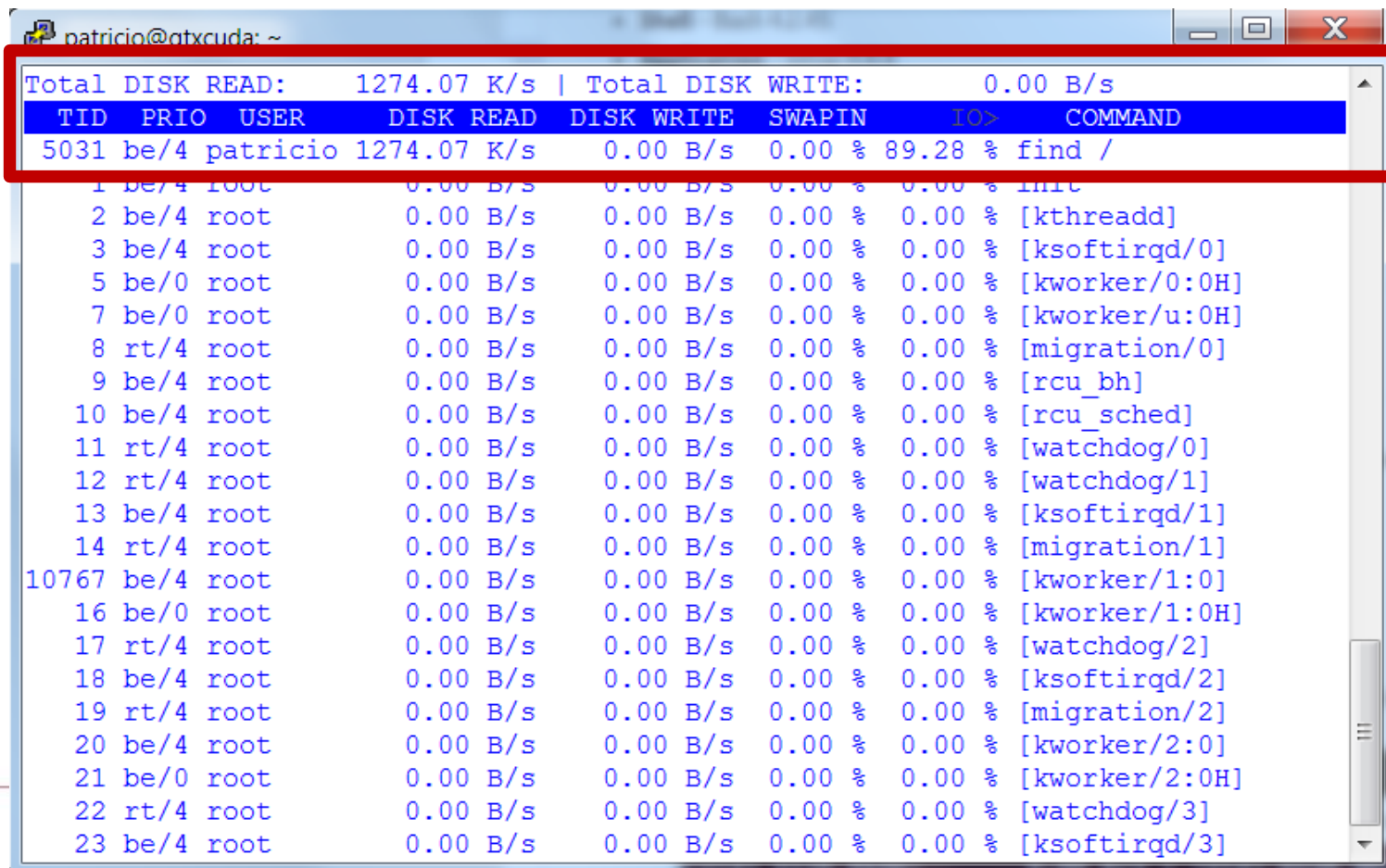
- ✓ Executam muitas operações de E/S e pouca computação
  - Tempo de execução é determinado pela velocidade dos dispositivos de E/S empregues
    - Processos E/S passam a maior parte do tempo na fila de processos bloqueados
  - Exemplos (aplicações que manipulam ficheiros/rede, etc.)
    - “find”, base de dados (tb podem ser CPU bound – sort), servidor de ficheiros, servidor WEB, etc.



# Processos dependentes de E/S (2)

## ✓ Utilitário `iostatop` (linux)

- Lista os processos consoante o respetivo uso de I/O
  - Exemplo: execução do comando `find /`

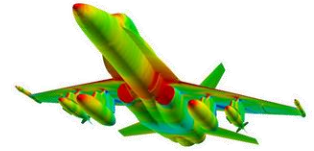


```
patricio@qtxcda: ~
Total DISK READ: 1274.07 K/s | Total DISK WRITE: 0.00 B/s
TID PRIO USER DISK READ DISK WRITE SWAPIN IO> COMMAND
5031 be/4 patricio 1274.07 K/s 0.00 B/s 0.00 % 89.28 % find /
1 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % init
2 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [kthreadd]
3 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [ksoftirqd/0]
5 be/0 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [kworker/0:0H]
7 be/0 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [kworker/u:0H]
8 rt/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [migration/0]
9 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [rcu_bh]
10 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [rcu_sched]
11 rt/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [watchdog/0]
12 rt/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [watchdog/1]
13 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [ksoftirqd/1]
14 rt/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [migration/1]
10767 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [kworker/1:0]
16 be/0 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [kworker/1:0H]
17 rt/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [watchdog/2]
18 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [ksoftirqd/2]
19 rt/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [migration/2]
20 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [kworker/2:0]
21 be/0 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [kworker/2:0H]
22 rt/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [watchdog/3]
23 be/4 root 0.00 B/s 0.00 B/s 0.00 % 0.00 % [ksoftirqd/3]
```

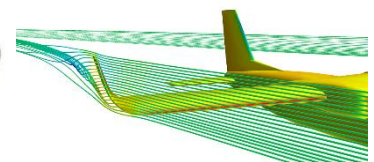
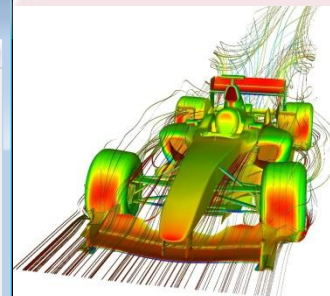
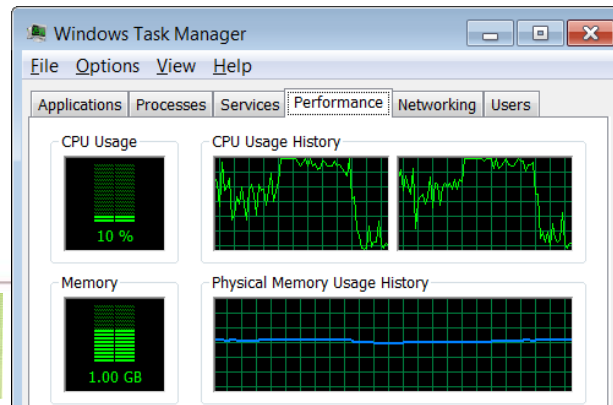
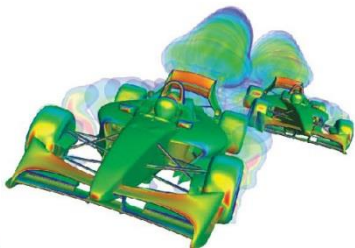
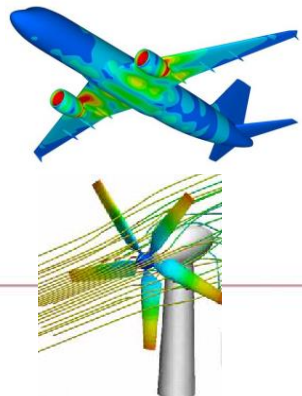


# Processos dependentes de CPU (1)

## ✓ Processos dependentes de CPU



- Executam muitas operações de computação “pura”
  - Processos dependentes do CPU passam a maior parte do tempo à espera do CPU
- O tempo de execução do processo está dependente da velocidade do CPU (e também da memória RAM)
  - Com um CPU mais rápida o processo executará mais rapidamente
- Exemplos: cálculo científico, renderização, etc.



# Processos dependentes de CPU (2)

## ✓ Utilitário `top`

– Exemplo: processo `nbench` a consumir todo o CPU



```

patricio@gtxcuda: ~
top - 15:50:51 up 177 days, 23:16, 2 users, load average: 0.81, 0.30, 0.14
Tasks: 121 total, 2 running, 119 sleeping, 0 stopped, 0 zombie
Cpu(s): 25.0%us, 0.1%sy, 0.0%ni, 74.9%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 3981976k total, 3519652k used, 462324k free, 508096k buffers
Swap: 3976188k total, 0k used, 3976188k free, 2484872k cached

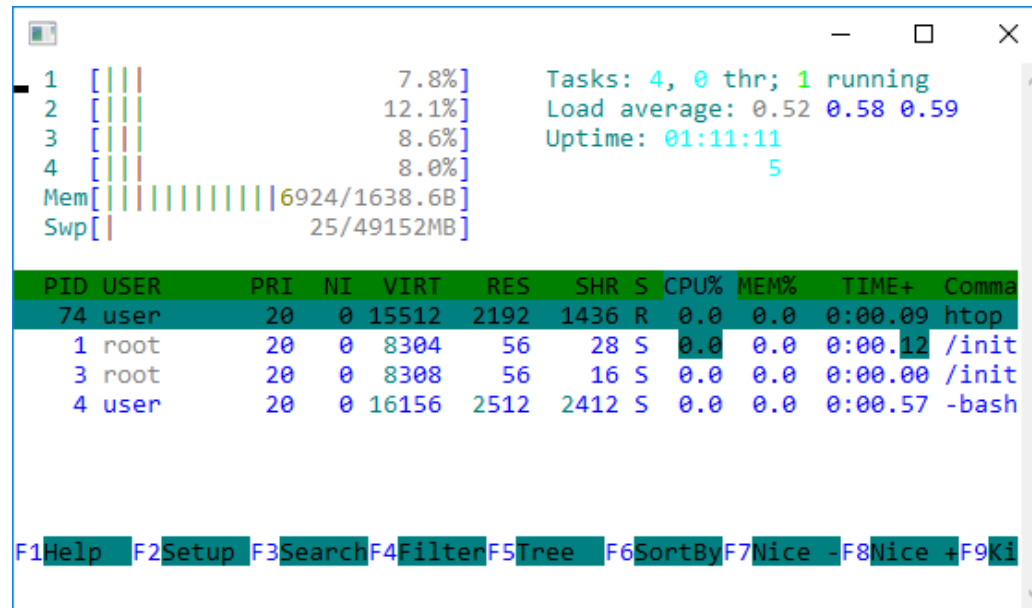
 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
 5314 patricio 20 0 1028 356 188 R 100 0.0 1:38.49 nbench
 11 root RT 0 0 0 0 S 0 0.0 0:53.11 watchdog/0
 5032 root 20 0 0 0 0 S 0 0.0 0:01.30 kworker/0:0
 1 root 20 0 24332 2044 1184 S 0 0.1 0:05.82 init
 2 root 20 0 0 0 0 S 0 0.0 0:00.19 kthreadd
 3 root 20 0 0 0 0 S 0 0.0 1:04.22 ksoftirqd/0
 5 root 0 -20 0 0 0 S 0 0.0 0:00.00 kworker/0:0H
 7 root 0 -20 0 0 0 S 0 0.0 0:00.00 kworker/u:0H
 8 root RT 0 0 0 0 S 0 0.0 4:42.87 migration/0
 9 root 20 0 0 0 0 S 0 0.0 0:00.00 rcu_bh
 10 root 20 0 0 0 0 S 0 0.0 1:57.41 rcu_sched
 12 root RT 0 0 0 0 S 0 0.0 0:51.40 watchdog/1
 13 root 20 0 0 0 0 S 0 0.0 1:33.61 ksoftirqd/1
 14 root RT 0 0 0 0 S 0 0.0 2:25.22 migration/1
 16 root 0 -20 0 0 0 S 0 0.0 0:00.00 kworker/1:0H
 17 root RT 0 0 0 0 S 0 0.0 0:35.29 watchdog/2
 18 root 20 0 0 0 0 S 0 0.0 0:53.01 ksoftirqd/2

```



# Sistema com pouca carga

- Utilitário htop
  - sudo apt install htop
  - Mostra a carga por core
    - 1,2,3,4
  - Permite interagir com os processos através das teclas de função (F1, F2,...)
- Exemplo
  - Load: 0.52
  - Cores: apenas core 2 > 10%
  - RAM: 6 GiB / 16 GiB

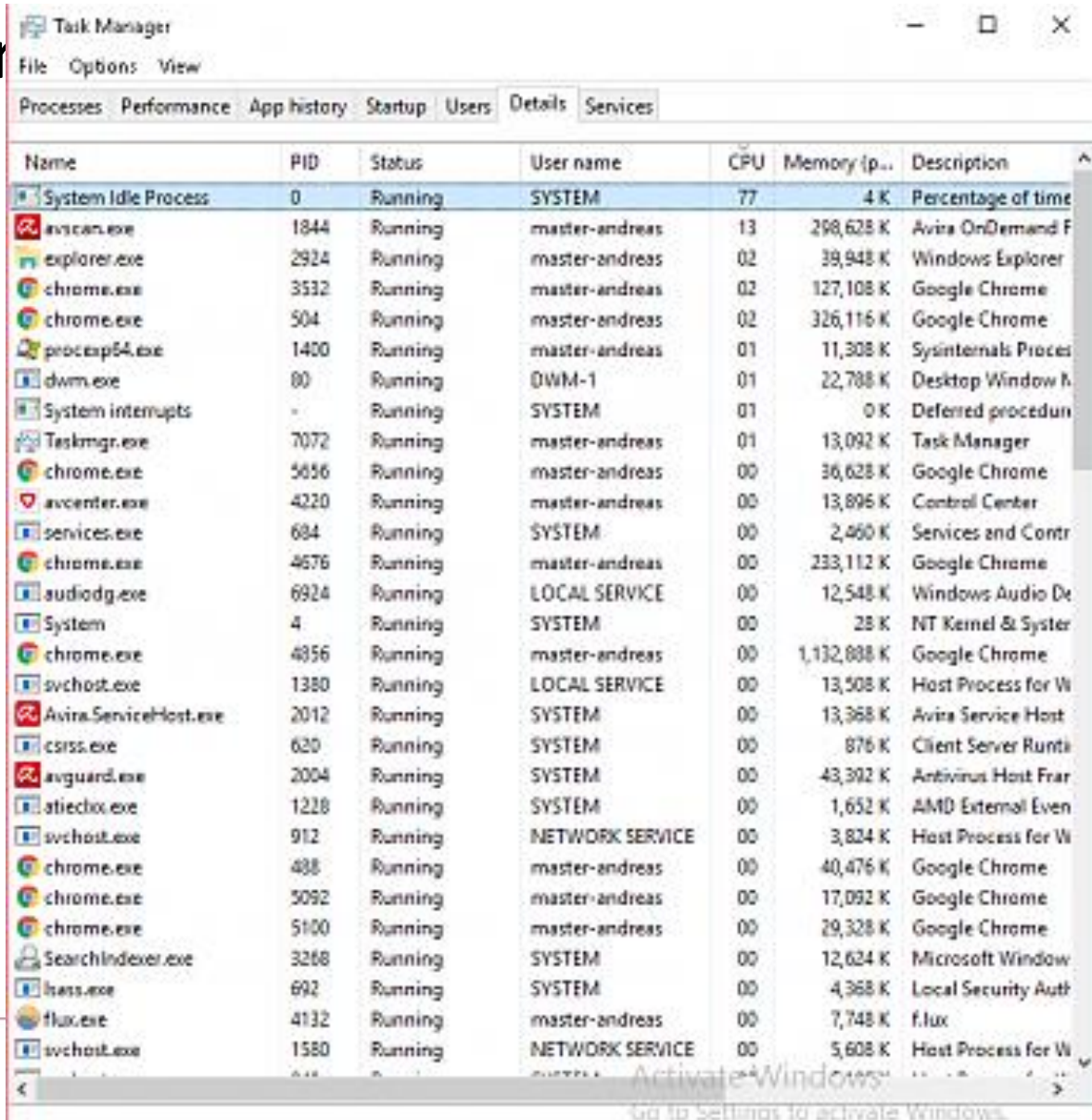


# O que é o *idle process* do Windows?

## ✓ Idle process

– Pr

o CPU



| Name                  | PID  | Status  | User name       | CPU | Memory (p... | Description         |
|-----------------------|------|---------|-----------------|-----|--------------|---------------------|
| System Idle Process   | 0    | Running | SYSTEM          | 77  | 4 K          | Percentage of time  |
| avscan.exe            | 1844 | Running | master-andreas  | 13  | 298,628 K    | Avira OnDemand F    |
| explorer.exe          | 2924 | Running | master-andreas  | 02  | 39,948 K     | Windows Explorer    |
| chrome.exe            | 3532 | Running | master-andreas  | 02  | 127,108 K    | Google Chrome       |
| chrome.exe            | 504  | Running | master-andreas  | 02  | 326,116 K    | Google Chrome       |
| proccsp64.exe         | 1400 | Running | master-andreas  | 01  | 11,308 K     | Sysinternals Proces |
| dwm.exe               | 80   | Running | DWM-1           | 01  | 22,788 K     | Desktop Window M    |
| System interrupts     | -    | Running | SYSTEM          | 01  | 0 K          | Deferred procedun   |
| Taskmgr.exe           | 7072 | Running | master-andreas  | 01  | 13,092 K     | Task Manager        |
| chrome.exe            | 5656 | Running | master-andreas  | 00  | 36,628 K     | Google Chrome       |
| avcenter.exe          | 4220 | Running | master-andreas  | 00  | 13,896 K     | Control Center      |
| services.exe          | 684  | Running | SYSTEM          | 00  | 2,460 K      | Services and Contr  |
| chrome.exe            | 4676 | Running | master-andreas  | 00  | 233,112 K    | Google Chrome       |
| audiodg.exe           | 6924 | Running | LOCAL SERVICE   | 00  | 12,548 K     | Windows Audio De    |
| System                | 4    | Running | SYSTEM          | 00  | 28 K         | NT Kernel & Syste   |
| chrome.exe            | 4856 | Running | master-andreas  | 00  | 1,132,888 K  | Google Chrome       |
| svchost.exe           | 1380 | Running | LOCAL SERVICE   | 00  | 13,508 K     | Host Process for W  |
| Avira.ServiceHost.exe | 2012 | Running | SYSTEM          | 00  | 13,368 K     | Avira Service Host  |
| csrss.exe             | 620  | Running | SYSTEM          | 00  | 876 K        | Client Server Runti |
| avguard.exe           | 2004 | Running | SYSTEM          | 00  | 43,392 K     | Antivirus Host Fram |
| atiecbox.exe          | 1228 | Running | SYSTEM          | 00  | 1,652 K      | AMD External Even   |
| svchost.exe           | 912  | Running | NETWORK SERVICE | 00  | 3,824 K      | Host Process for W  |
| chrome.exe            | 488  | Running | master-andreas  | 00  | 40,476 K     | Google Chrome       |
| chrome.exe            | 5092 | Running | master-andreas  | 00  | 17,092 K     | Google Chrome       |
| chrome.exe            | 5100 | Running | master-andreas  | 00  | 29,328 K     | Google Chrome       |
| SearchIndexer.exe     | 3268 | Running | SYSTEM          | 00  | 12,624 K     | Microsoft Window    |
| lsass.exe             | 692  | Running | SYSTEM          | 00  | 4,368 K      | Local Security Auth |
| flux.exe              | 4132 | Running | master-andreas  | 00  | 7,748 K      | f.lux               |
| svchost.exe           | 1580 | Running | NETWORK SERVICE | 00  | 5,608 K      | Host Process for W  |



- ✓ “system idle process”
  - Processo que executa quando...o SO não tem mais nenhum processo que possa ocupar o CPU
  - A percentagem de CPU consumida pelo “system idle process” é um bom indicador da taxa de uso da máquina
    - Percentagem elevada de “idle” significa que o processador está com pouco uso
      - É típico em máquinas de uso interactivo, que a percentagem de idle seja próxima dos 95%
  - O “system idle process” executa a instrução HALT, que têm um consumo muito reduzido de energia
  - Designação em língua portuguesa de “idle process”
    - processo inativo do sistema

**IPL**escola superior de tecnologia e gestão  
instituto politécnico de leiria

# Assustado com o *idle* process...

Windows 10 Forums » Performance &amp; Maintenance »

**[System Idle Process] What is it? why does it use a ton of CPU?**

08 Feb 2017



The screenshot shows two windows side-by-side. The left window is 'Process Explorer' from Sysinternals, displaying a list of processes. The 'System Idle Process' is at the top, showing 77.87% CPU usage, 0 K private bytes, and 4 K working set. The right window is 'Task Manager', showing the same process list. The 'System Idle Process' is also at the top, showing 77% CPU usage. The status bar at the bottom of Process Explorer indicates 'CPU Usage: 22.13% | Commit Charge: 65.67% | Processes: 84 | Physical Usage: 56.77%'.

| Process             | CPU    | Private Bytes | Working Set | PID  | Description                      | Company Name     |
|---------------------|--------|---------------|-------------|------|----------------------------------|------------------|
| System Idle Process | 77.87  | 0 K           | 4 K         | 0    |                                  |                  |
| svchost.exe         | 10.22  | 305,672 K     | 332,836 K   | 1844 |                                  |                  |
| chrome.exe          | 1.53   | 367,548 K     | 372,144 K   | 504  | Google Chrome                    | Google Inc.      |
| process64.exe       | 1.39   | 14,900 K      | 30,908 K    | 1400 | Sysinternals Process Explorer    | Sysinternals - v |
| chrome.exe          | 1.34   | 144,872 K     | 154,336 K   | 3532 | Google Chrome                    | Google Inc.      |
| dwm.exe             | 0.89   | 47,536 K      | 51,132 K    | 80   |                                  |                  |
| Taskmgr.exe         | 0.86   | 16,044 K      | 37,566 K    | 7072 |                                  |                  |
| chrome.exe          | 0.77   | 56,308 K      | 75,476 K    | 5056 | Google Chrome                    | Google Inc.      |
| Interrupts          | 0.74   | 0 K           | 0 K         | n/a  | Hardware Interrupts and DPCs     |                  |
| System              | 0.67   | 128 K         | 128 K       | 4    |                                  |                  |
| audiodg.exe         | 0.60   | 18,624 K      | 23,292 K    | 6924 |                                  |                  |
| chrome.exe          | 0.58   | 256,800 K     | 305,676 K   | 4676 | Google Chrome                    | Google Inc.      |
| explorer.exe        | 0.40   | 52,964 K      | 106,324 K   | 2924 | Windows Explorer                 | Microsoft Corp   |
| avcenterd.exe       | 0.43   | 15,728 K      | 37,032 K    | 4220 | Control Center                   | Avira Operator   |
| csrss.exe           | 0.35   | 4,964 K       | 4,280 K     | 620  |                                  |                  |
| avgguard.exe        | 0.33   | 329,392 K     | 62,076 K    | 2004 | Avira Host Framework Se...       | Avira Operator   |
| chrome.exe          | 0.29   | 1,313,140 K   | 1,162,960 K | 4056 | Google Chrome                    | Google Inc.      |
| svchost.exe         | 0.14   | 15,512 K      | 26,400 K    | 1380 | Host Process for Windows S...    | Microsoft Corp   |
| svchost.exe         | 0.10   | 2,716 K       | 9,584 K     | 1428 | Host Process for Windows S...    | Microsoft Corp   |
| taskhostw.exe       | 0.09   | 3,736 K       | 14,500 K    | 3952 | Host Process for Windows T...    | Microsoft Corp   |
| schex.exe           | 0.09   | 7,420 K       | 4,656 K     | 1804 | Avira Host Framework Se...       | Avira Operator   |
| services.exe        | 0.04   | 2,852 K       | 6,640 K     | 604  |                                  |                  |
| chrome.exe          | 0.04   | 45,840 K      | 51,912 K    | 5108 | Google Chrome                    | Google Inc.      |
| avscan.exe          | 0.04   | 2,284 K       | 9,876 K     | 1228 |                                  |                  |
| svchost.exe         | 0.03   | 4,392 K       | 9,636 K     | 912  | Host Process for Windows S...    | Microsoft Corp   |
| flux.exe            | 0.03   | 16,940 K      | 22,908 K    | 4132 | Flux Software L                  |                  |
| chrome.exe          | 0.02   | 161,276 K     | 207,524 K   | 972  | Google Chrome                    | Google Inc.      |
| chrome.exe          | 0.01   | 125,008 K     | 138,976 K   | 5980 | Google Chrome                    | Google Inc.      |
| avscan.exe          | < 0.01 | 4,400 K       | 15,436 K    | 7112 | Avira OnDemand File Scanner      | Avira Operator   |
| avndv.exe           | < 0.01 | 170,184 K     | 6,632 K     | 4308 |                                  |                  |
| chrome.exe          | < 0.01 | 57,560 K      | 68,392 K    | 488  | Google Chrome                    | Google Inc.      |
| lsass.exe           | < 0.01 | 5,176 K       | 14,500 K    | 692  | Local Security Authority Proc... | Microsoft Corp   |
| chrome.exe          | < 0.01 | 27,172 K      | 34,568 K    | 2824 | Google Chrome                    | Google Inc.      |
| chrome.exe          | < 0.01 | 31,376 K      | 39,148 K    | 5092 | Google Chrome                    | Google Inc.      |
| chrome.exe          | < 0.01 | 25,908 K      | 33,888 K    | 5116 | Google Chrome                    | Google Inc.      |
| chrome.exe          | < 0.01 | 25,724 K      | 32,944 K    | 4104 | Google Chrome                    | Google Inc.      |
| chrome.exe          | < 0.01 | 45,724 K      | 52,708 K    | 5100 | Google Chrome                    | Google Inc.      |
| svchost.exe         | < 0.01 | ...           | ...         | ...  | ...                              | ...              |

Fonte: <https://bit.ly/2BHR1xi>



# What's in a process?

JULIA EVANS  
@b0rk

## PID

process #129  
reporting for  
duty!



## USER and GROUP

who ran  
you?

julia!



## ENVIRONMENT VARIABLES

like PATH! you  
can set them with:  
`$ env A=val ./program`

## SIGNAL HANDLERS



I ignore  
SIGTERM!

I shut  
down safely!



## WORKING DIRECTORY

Relative paths (./blah)  
are relative to the  
working directory!  
`chdir` changes it.

## PARENT PID



PID 1 (init)  
is everyone's  
ancestor



PID 147



PID 129

## COMMAND LINE ARGUMENTS

See them in  
`/proc/PID/cmdline`

## OPEN FILES

Every open file has  
an offset.



I've read 8000  
bytes of that one

## MEMORY

heap! stack!   
shared libraries!  
the program's binary!  
mmaped files!

## THREADS

sometimes one  
sometimes LOTS

## CAPABILITIES



I have  
CAP\_PTRACE

well I have  
CAP\_SYS\_ADMIN

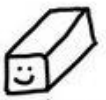


## NAMESPACES



I'm in the host  
network namespace

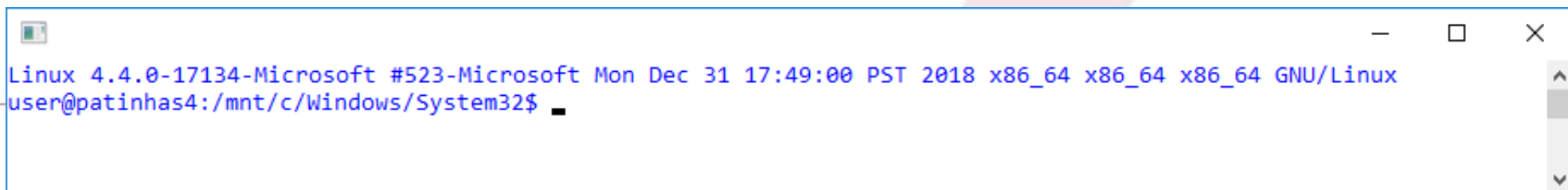
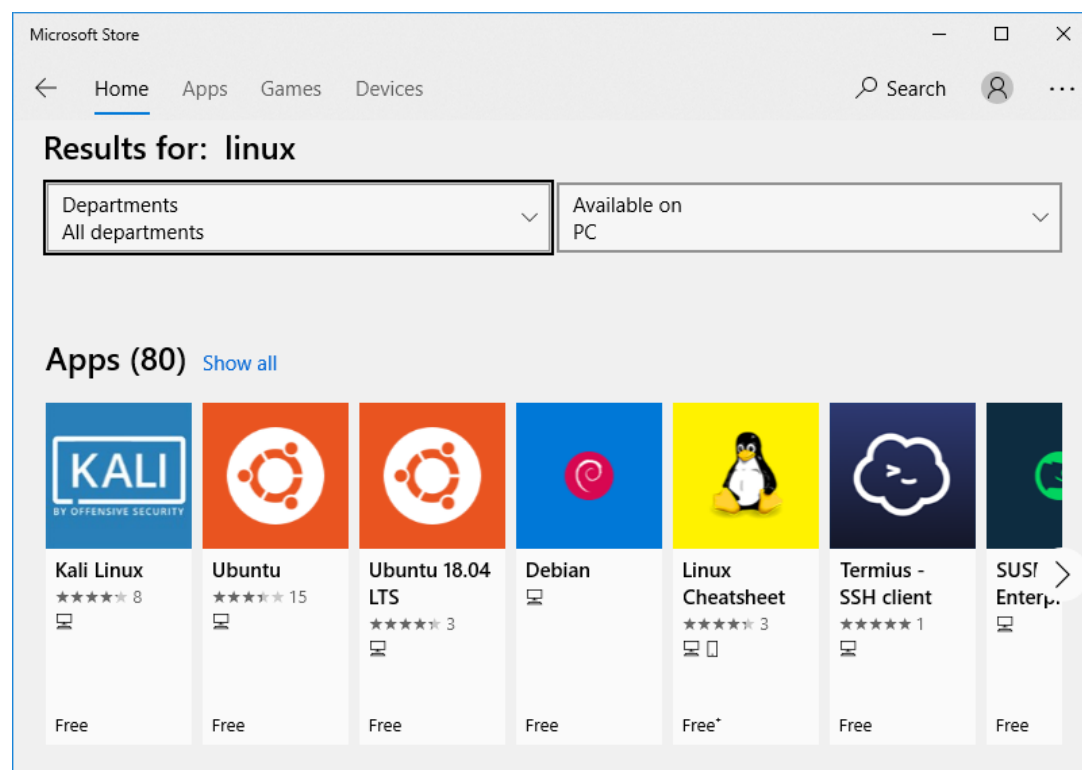
I have my own  
namespace!



container  
process

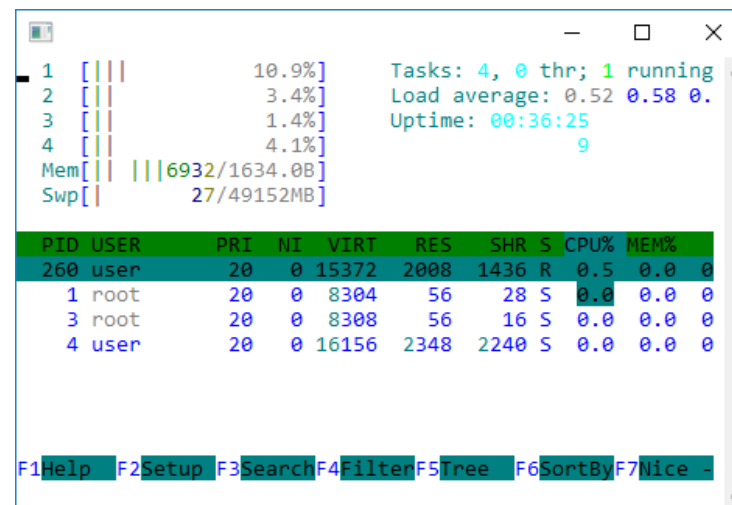
# WSL: caso de estudo (#1)

- Windows Subsystem Linux (WSL)
  - Sistema de compatibilidade para suporte a aplicações GNU/Linux
  - Disponível para Windows 10/64 bits (Windows Store)
    - Instalação
      - <https://docs.microsoft.com/en-us/windows/wsl/install-win10>



# WSL: caso de estudo (#2)

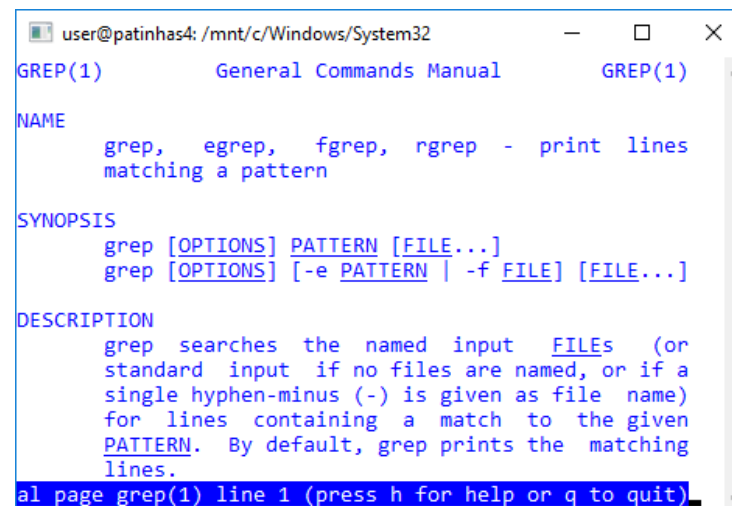
- Suporta os comandos/utilitários do modo utilizador do linux
  - ls,ps,cut,grep,pwd,top,tr,man,...
  - Programas podem ser instalados/removidos consoante a distribuição
    - Exemplos
- sudo apt install vim
- sudo apt show gcc
- sudo apt remove htop



Tasks: 4, 0 thr; 1 running  
Load average: 0.52 0.58 0.  
Uptime: 00:36:25  
9

| PID | USER | PRI | NI | VIRT  | RES  | SHR  | S | CPU% | MEM% |
|-----|------|-----|----|-------|------|------|---|------|------|
| 260 | user | 20  | 0  | 15372 | 2008 | 1436 | R | 0.5  | 0.0  |
| 1   | root | 20  | 0  | 8304  | 56   | 28   | S | 0.0  | 0.0  |
| 3   | root | 20  | 0  | 8308  | 56   | 16   | S | 0.0  | 0.0  |
| 4   | user | 20  | 0  | 16156 | 2348 | 2240 | S | 0.0  | 0.0  |

F1 Help F2 Setup F3 Search F4 Filter F5 Tree F6 SortBy F7 Nice



user@patinhas4: /mnt/c/Windows/System32

GREP(1) General Commands Manual GREP(1)

NAME

grep, egrep, fgrep, rgrep - print lines matching a pattern

SYNOPSIS

grep [OPTIONS] PATTERN [FILE...]  
grep [OPTIONS] [-e PATTERN | -f FILE] [FILE...]

DESCRIPTION

grep searches the named input FILES (or standard input if no files are named, or if a single hyphen-minus (-) is given as file name) for lines containing a match to the given PATTERN. By default, grep prints the matching lines.

al page grep(1) line 1 (press h for help or q to quit)

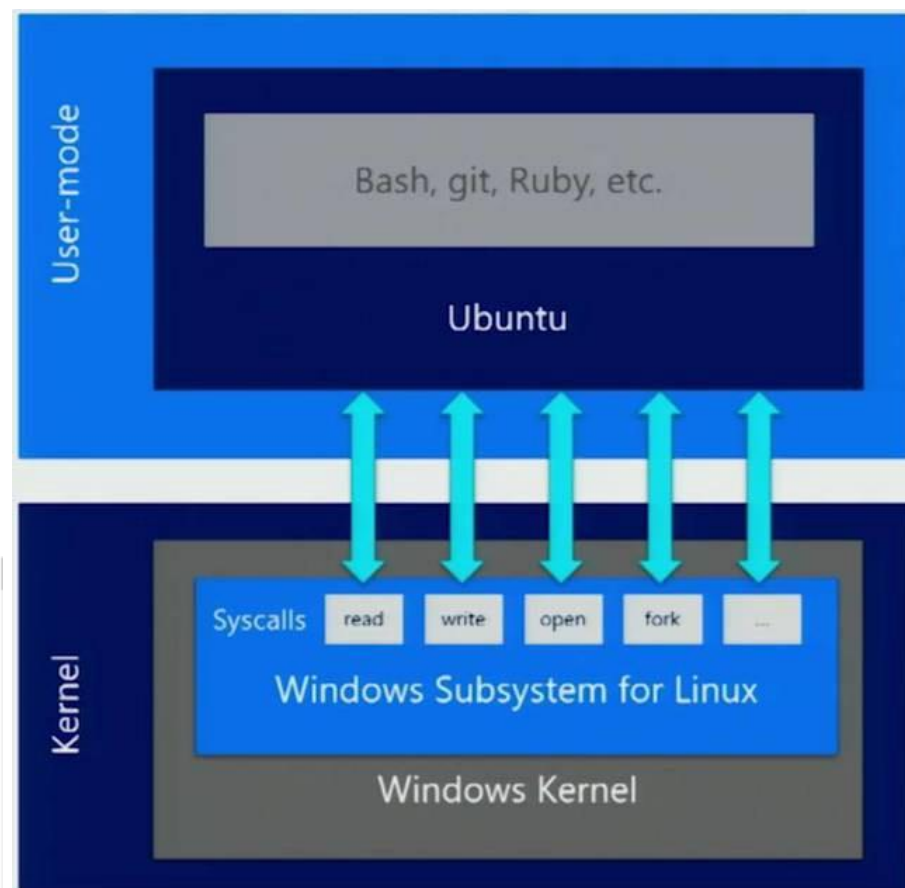
# WSL: caso de estudo (#3)

- Arquitetura
  - Serviço LXSS Manager interage com o subsistema Linux
    - Responsável por tratar das chamadas ao sistema do Linux

```
c:\>sc query lxss

SERVICE_NAME: lxss
 TYPE : 1 KERNEL_DRIVER
 STATE : 4 RUNNING
 (STOPPABLE, NOT_PAUSABLE, IGNORES_SHUT
DOWN)
 WIN32_EXIT_CODE : 0 (0x0)
 SERVICE_EXIT_CODE : 0 (0x0)
 CHECKPOINT : 0x0
 WAIT_HINT : 0x0
```

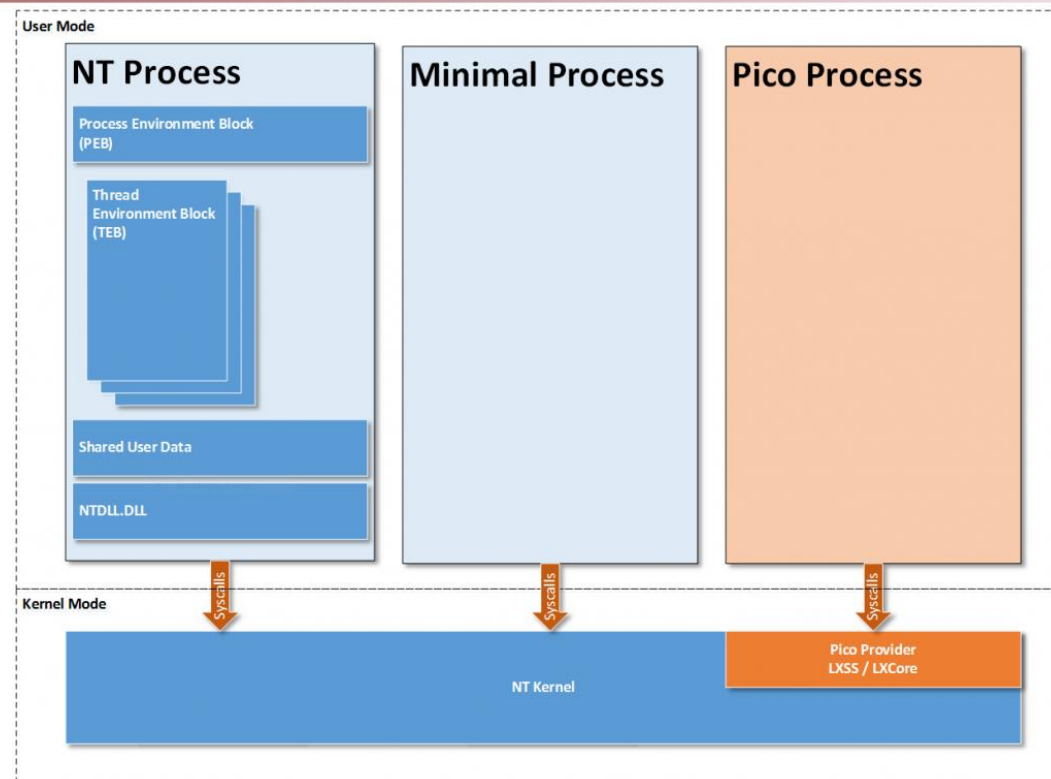
Serviço LXSS



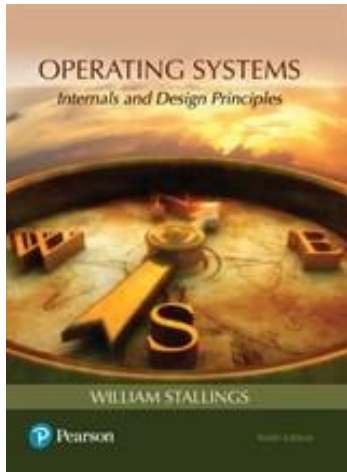


# WSL: caso de estudo (#4)

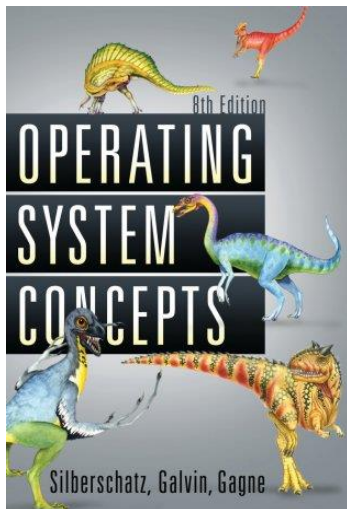
- WSL faz uso de *pico processos*
  - Pico processo
    - Processo mínimo cujos pedidos são processados por um “pico provider”
    - Pico provider é um módulo do núcleo (*kernel*) do SO
- WSL implementa case-sensitiveness no sistema de ficheiros
  - maiúscula ≠ minúscula
  - Windows é case-insensitive



- (algumas) Limitações
  - Apenas aplicações não dependentes de serviços do kernel do Linux
  - Nem todos os diretórios podem ser acedidos



- Capítulo 3 de “Operating Systems – Internals and Design Principles”, William Stallings, 9ª edição, 2017



- Capítulo 3 (3.1 a 3.3) de “Operating Systems Concepts”, A. Silberschatz, 8ª edição, 2011 (ISBN:1118112733)



- Entrevista Linus Torvalds – TED 2016  
[https://www.ted.com/talks/linus\\_torvalds\\_the\\_mind\\_behind\\_linux](https://www.ted.com/talks/linus_torvalds_the_mind_behind_linux)

