

# Estruturas do SO

Alberto Felipe Friderichs Barros

## Primeira geração (1945 — 1955)

- Tinha com principal característica o uso de **válvulas eletrônicas**, possuindo dimensões enormes.
- Utilizavam quilômetros de fios, chegando a atingir temperaturas muito elevadas, o que frequentemente causava problemas de funcionamento.
- Várias máquinas, ENIAC mais famosa de todas.
- **Ausência de SO: Programação feita em linguagem de máquina.**

## ENIAC

25 metros de comprimento por 5,50 metros de altura.  
O seu peso total era de 30 toneladas.



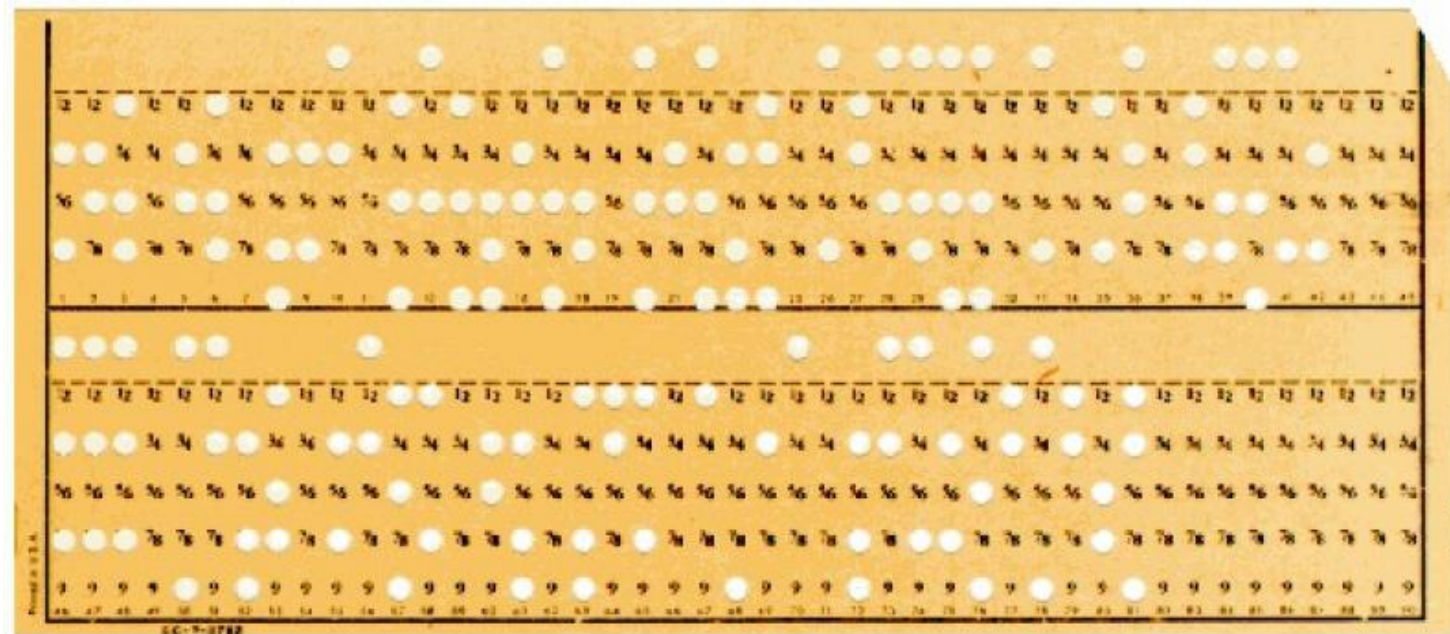
## Segunda geração (1955 — 1965)

- Houve a substituição das válvulas eletrônicas por **transistores**, o que diminuiu em muito tamanho do hardware.
- Uso comercial.
- O IBM 7030 foi o primeiro supercomputador lançado. Tamanho era bem reduzido comparado ao ENIAC, podendo ocupar somente uma sala comum.
- Mais rápido e confiável.

- Ele era utilizado por grandes companhias, custando em torno de 13 milhões de dólares.
- Surgiu o conceito de sistema operacional do tipo **Batch (lote)**.
- Cada máquina possuía seu sistema específico.
- Surgimento das primeiras linguagens de programação de alto nível – Fortran (IBM).



Vários comandos já poderiam ser executado em sequências através de um cartão perfurado, eliminando parte do trabalho do operador de terminal. Normalmente um programa era composto por um conjunto de cartões inseridos na ordem correta pelo usuário do sistema. **Alto Custo.**





## Surgimento dos Sistemas Operacionais

### Multics:

Tentativa de um sistema operacional foi entre 1964 e 1965 não era muito funcional. Criado por três empresas, porém essas empresas romperam pois cada uma queria fazer algo diferente.



	NOV	DEC	TOTAL
HOME BUDGET, 1979			
MONTH			
SALARY	2500.00	2500.00	30000.00
OTHER			
INCOME	2500.00	2500.00	30000.00
FOOD	400.00	400.00	4800.00
RENT	350.00	350.00	4200.00
HEAT	110.00	120.00	575.00
REC.	100.00	100.00	1200.00
TAXES	1000.00	1000.00	12000.00
ENTERTAIN	100.00	100.00	1200.00
MISC	100.00	100.00	1200.00
CAR	300.00	300.00	3600.00
EXPENSES	2460.00	2470.00	28775.00
REMAINDER	40.00	30.00	1225.00
SAVINGS	30.00	30.00	3600.00

## Terceira geração (1965 — 1980).

- Os computadores desta geração foram conhecidos pelo uso de **circuitos integrados**.
- As máquinas se tornaram mais velozes, com um número maior de funcionalidades. O preço também diminuiu consideravelmente. Diminuição de tamanho.
- Multiprogramação. Multitarefa e multiusuário.

**1969:** Ken Thompson começou a reescrever o que ficou do multics. E foi criado o **UNIX**.



**1973:** UNIX foi reescrito, fazendo com que ele pudesse ser distribuído para mais organizações comerciais.

**1977:** foi lançado BSD um SO fortemente baseado no UNIX focado principalmente para execução em máquinas de especificas de alto desempenho. Usado em computadores de grande porte.



```
Booting from Floppy...
Seek error 128, req = 0, at = -1
unit 0, type 0, sectrac 18, blknum 17
386BSD Release 0.1 by William and Lynne Jolitz. [0.1.24 07/14/92 19:07]
Copyright (c) 1989,1990,1991,1992 William F. Jolitz. All rights reserved.
Based in part on work by the 386BSD User Community and the
BSD Networking Software, Release 2 by UCB EECS Department.
pc0<color> at 0x60 irq 1 on isa
com1 fifo at 0x3f8 irq 4 on isa
com2 fifo at 0x2f8 irq 3 on isa
wd0 <ST1102AT> at 0x1f0 irq 14 on isa
fd0 drives 0: 1.2M at 0x3f0 irq 6 drq 2 on isa
ne0 ethernet address fe:fd:de:ad:be:ef at 0x300 irq 9 on isa
np0 at 0xf0 irq 13 on isa
changing root device to fd0a

warning: no swap space present (yet)
386BSD Distribution Installation Floppy (Tiny 386BSD) Release 0.1

Please read the installation notes (type 'zmore INSTALL.NOTES')
and registration information (type 'more REGISTRATION') before use.
To install on hard disk drive, type 'install'.

erase ^?, werase ^H, kill ^U, intr ^C
#
```



**1975:** Bil Gates fundou a Microsoft, com o objetivo de desenvolver softwares.

**1977:** jovens programadores tinham uma ideia “absurda”, criar sistemas operacionais para o uso de pessoas comuns e um dos primeiros a pensar desta forma absurda foi o Steve Jobs o fundador da Apple desde a criação. Alguns anos depois, começou a desenvolver seu próprio SO, com interface gráfica totalmente revolucionário.



- Piratas do vale do silício: <https://www.youtube.com/watch?v=vSeiYLv4-so>
- Conta a história dos fundadores da Apple e da Microsoft.
- Até 1980, a informática era algo distante, que não fazia parte do universo das pessoas comuns. Os dois, ainda estudantes, lideraram uma revolução que integrou os computadores ao nosso dia a dia.



## Quarta geração (1980 - Atual)

- Começo da era dos computadores pessoais, **microcomputadores**.
- Computadores menores. **Interface gráfica**.
- Microprocessadores tornou a informática mais acessível.
- Melhor desempenho e maior capacidade.
- Tornou-se mais barato.
- Processadores com múltiplos núcleos.







**INSTITUTO  
FEDERAL**  
Santa Catarina







INSTITUTO  
FEDERAL  
Santa Catarina

**CONTROL**  
TECH INC



## Quarta geração (1980 - Atual)

- **1982:** MsDOS, interface baseado em modo texto.
- **1985:** Bil Gates apresenta o Windows 1.0, roda MsDOS em uma interface gráfica.

```
>dir /w

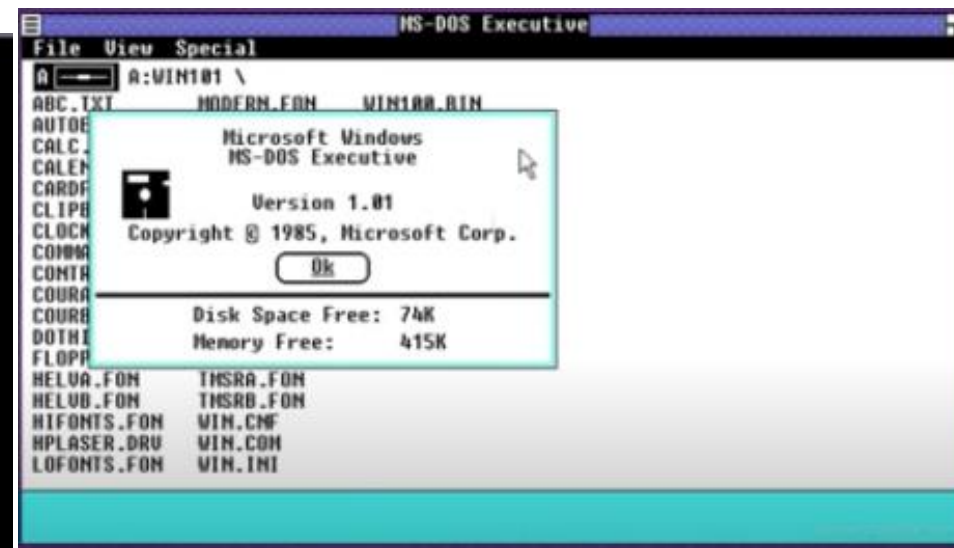
Volume in drive A has no label
Volume Serial Number is BB3C-98D8
Directory of A:\

MZ100.DOC_    AVEXTRA.TXT_    COMDEV.IN_     DEPCA.DOC_     E20ND.DOC_
Z1ND.DOC_     ELNK.DOC_       ELNK16.DOC_    ELNK3.DOC_     ELNK11.DOC_
LNKMC.DOC_    ELNKPL.DOC_     EXP16.DOC_     EXPAND.EXE     I82593.DOC_
BMTOK.DOC_    IFSHLP.SY_      LICENSE.TXT_    LM21DRV.UP_    MSDLC.EX_
DIS39XR.DOC_  NDISHLP.SY_     NE1000.DOC_    NE2000.DOC_    NET.EX_
ET.MS_        NETBIND.COM_    NETH.MS_       NI6510.DOC_    NWLINK.EXE
EMDLC.INF_    OEMODI.IN_      OEMRAS.IN_     OEMTCPIP.INF_  OLITOK.DOC_
E2NDIS.DOC_   PENDIS.DOC_     PRO4.DOC_      PRORAPM.DOC_   PROTMAN.DOC_
ROTHAN.EX_    RASCOPY.BA_     README.TXT_    SETUP.EXE      SMCMAC.DOC_
MC_ARC.DOC_   STRM.DOC_       TLNK.DOC_      UCNET.INF_     UCSETUP.INF_
CSYS.INI_     WORKGRP.SY_

52 file(s)      1,138,638 bytes
                  312,832 bytes free

>
>
```

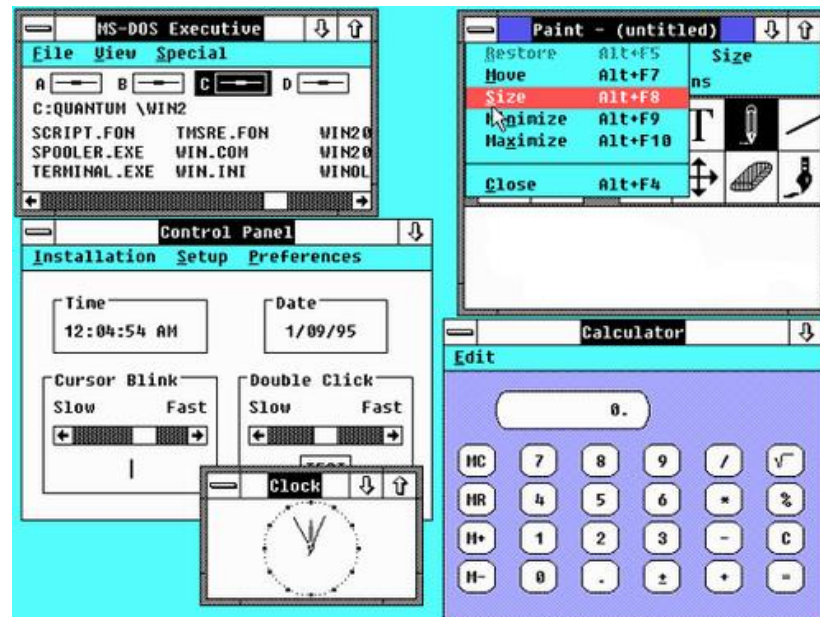
MsDOS



Windows 1.0

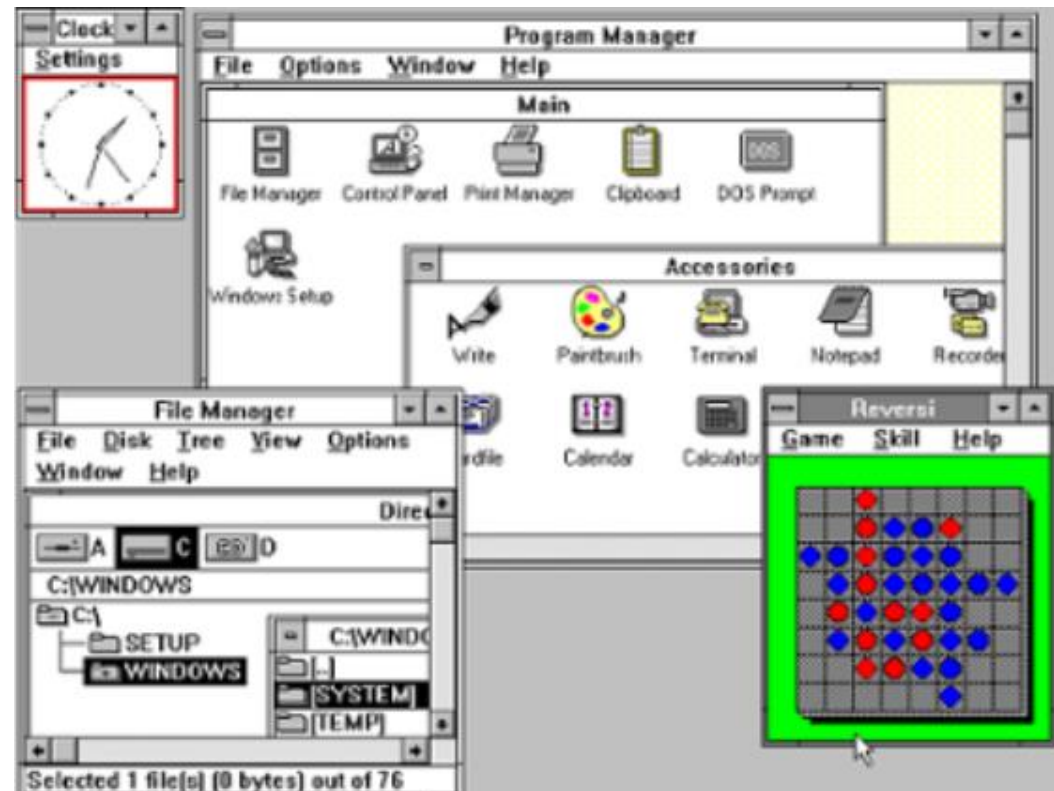
## Quarta geração (1980 - Atual)

- Final dos anos 90, a **Apple perdeu bastante mercado** do seu SO para a Microsoft, pois apresentava muitos problemas.
- 1987: Windows 2.0



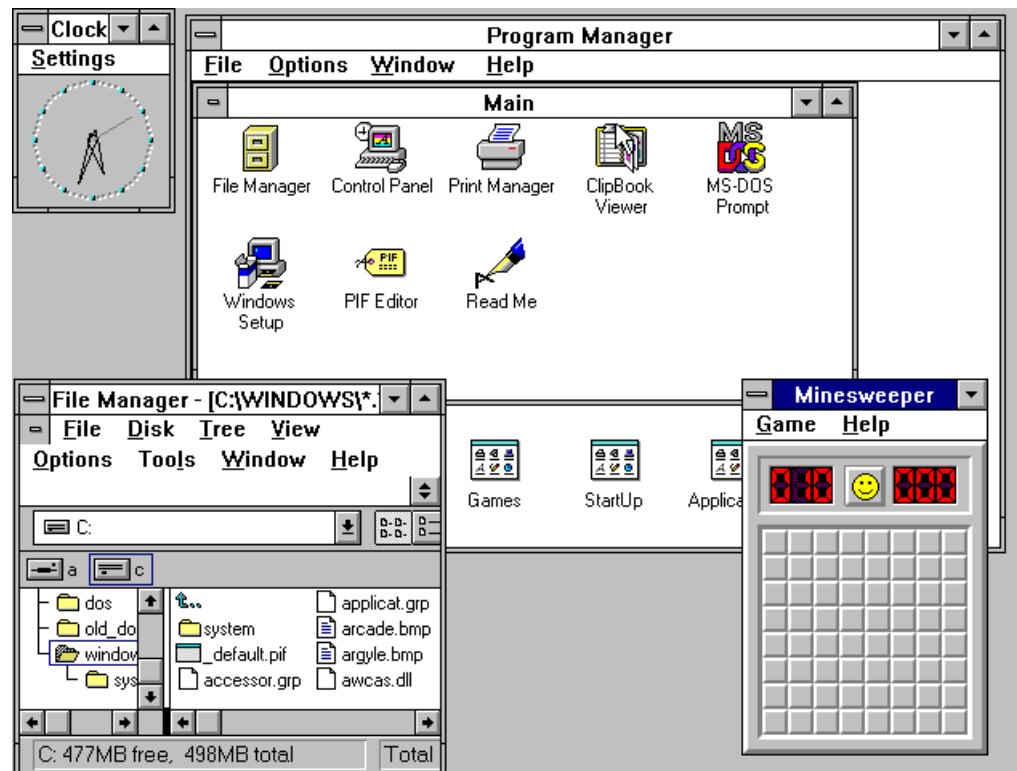
## Quarta geração (1980 - Atual)

- 1990: lançado o windows 3.0. O suporte a 256 cores e o jogo “Paciência” foram novidades importantes dessa versão do SO.



## Quarta geração (1980 - Atual)

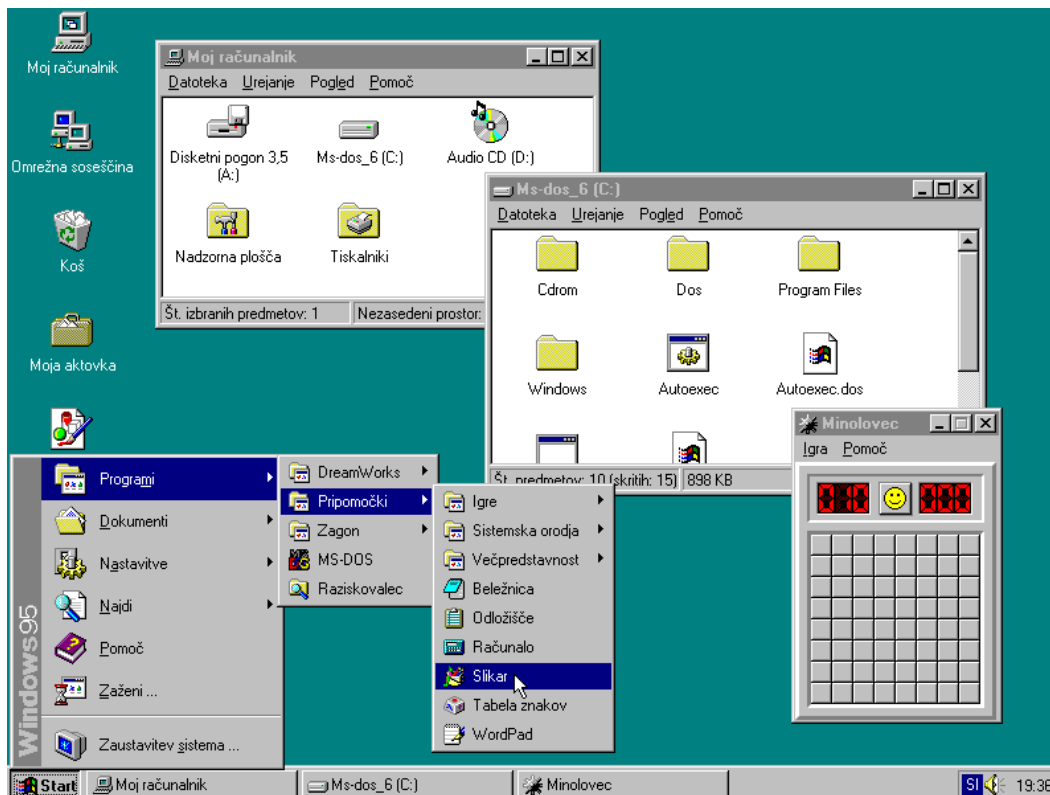
- Windows 3.1 exigia 1 MB de memória RAM para ser executado e, depois de instalado, ocupava apenas 15 MB do disco rígido. O jogo “Campo Minado” fez a sua estreia nesta versão do sistema operacional da Microsoft.





## Quarta geração (1980 - Atual)

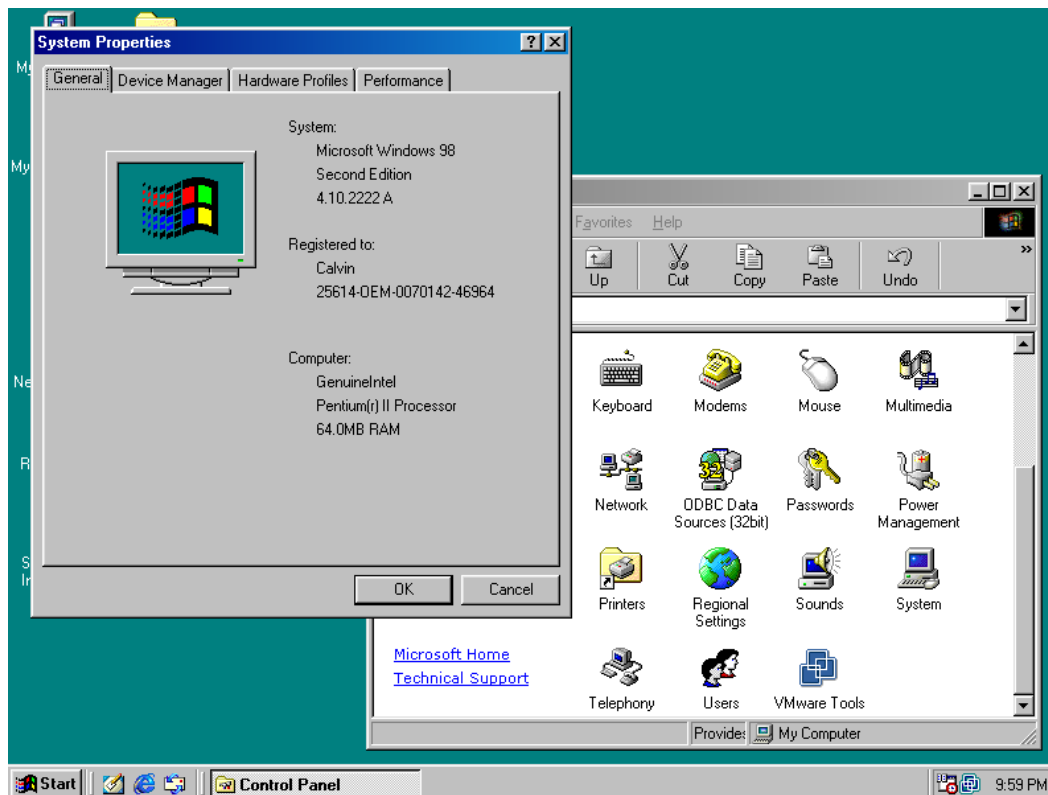
- 1995: Windows 95, trouxe, pela primeira vez, o Menu Iniciar e a Barra de Ferramentas tão familiares para todos nós. Foi nesta versão que o Internet Explorer fez a sua estreia.





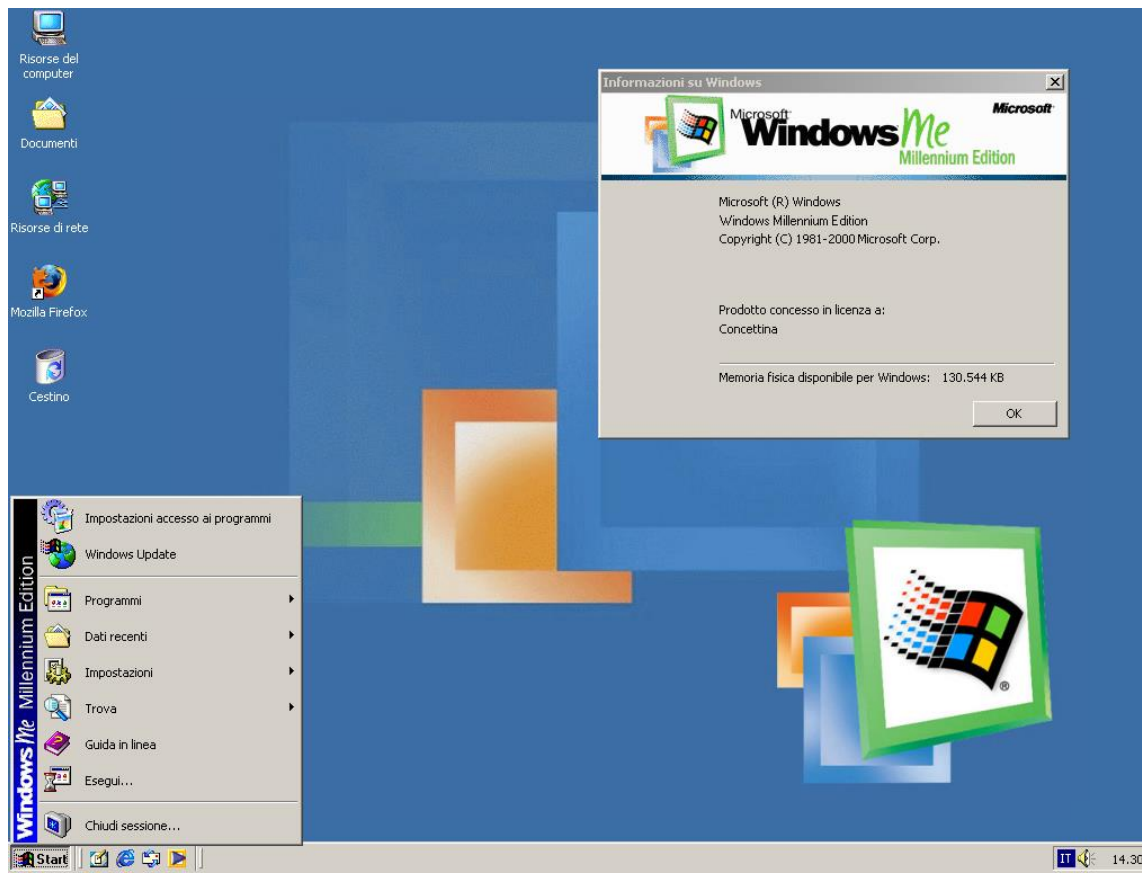
## Quarta geração (1980 - Atual)

- 1998: Windows 98, introduziu o recurso de avançar e voltar na navegação, além da barra de endereço no Windows Explorer. O **suporte ao padrão USB** também foi bastante aprimorado, dando início a uma adoção generalizada desse formato.



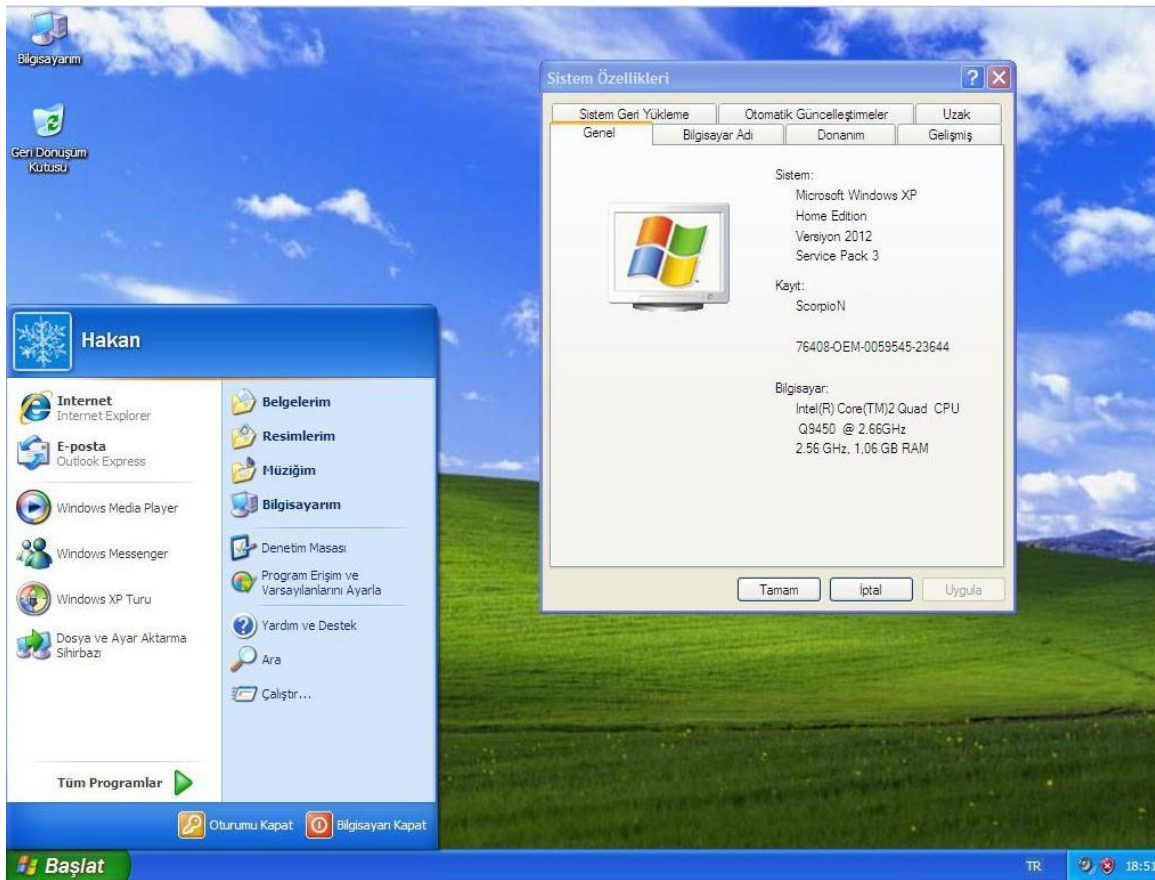
## Quarta geração (1980 - Atual)

- 2000: O Windows Millennium Edition foi a última versão do SO baseada no MS-DOS e considerada por muitos como a pior de todas. Apresentava bugs e problemas de instalação.



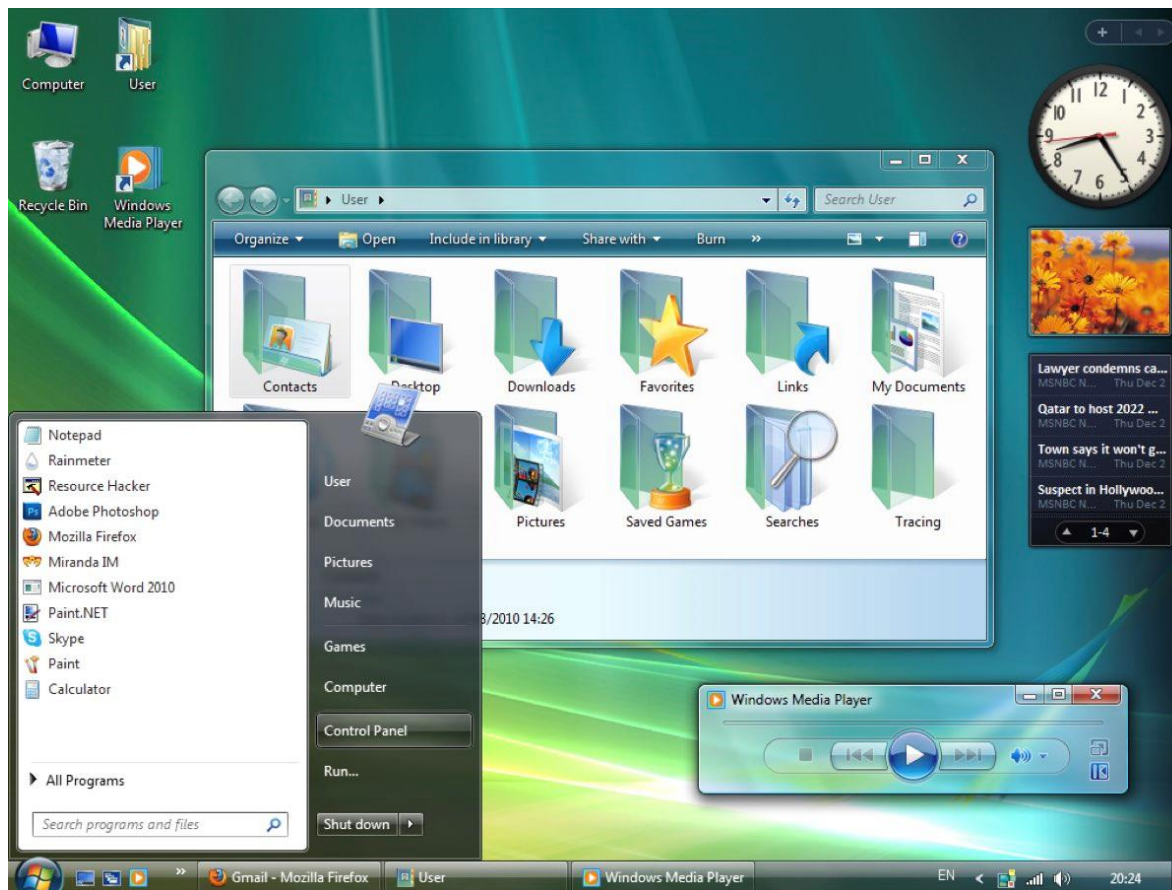
## Quarta geração (1980 - Atual)

- 2001: Windows XP, alguns argumentam que esta seja a melhor versão do sistema operacional da Microsoft, sendo o que mais durou no mercado, recebendo suporte até o mês de abril de 2014



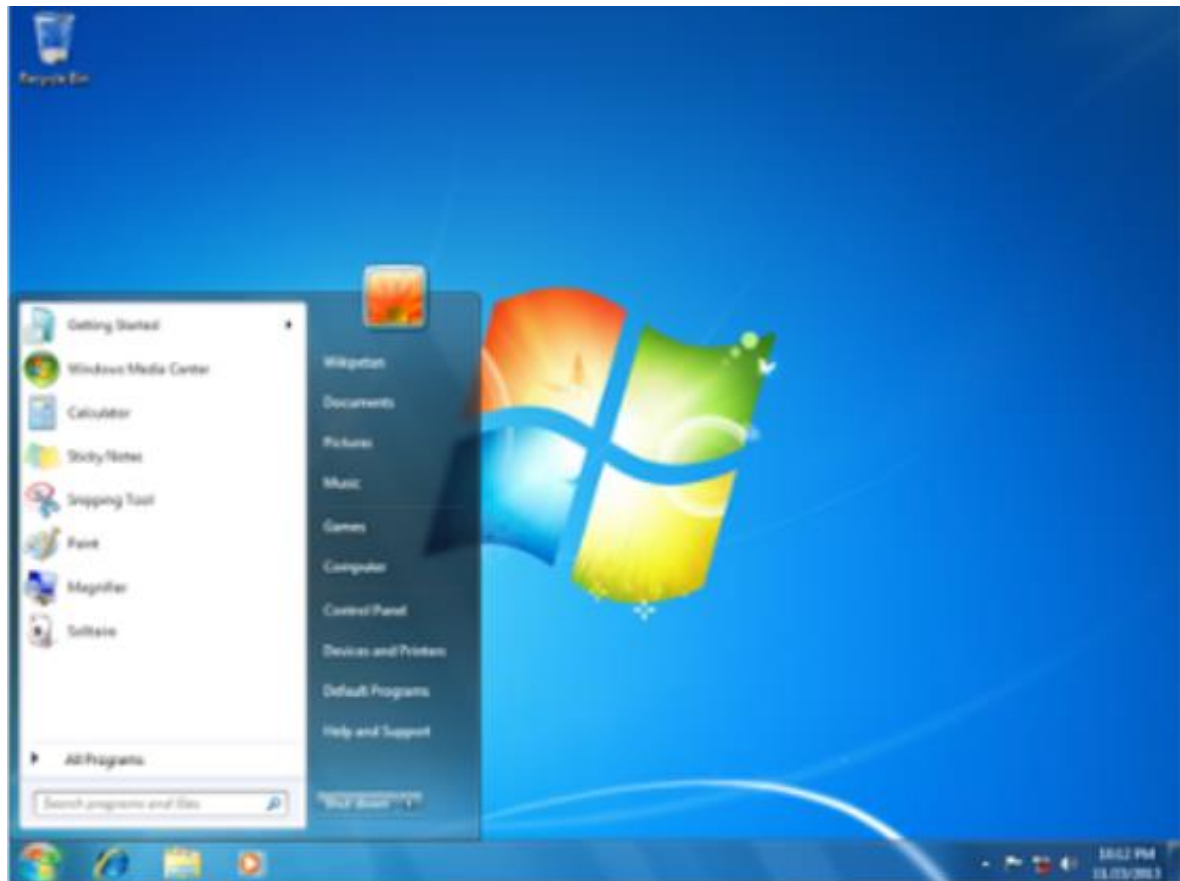
## Quarta geração (1980 - Atual)

- 2007: Windows Vista, o sistema apresentava um visual moderno, porém uma série de problemas e funcionalidades mal implementadas. Além de exigir muito do hardware.



## Quarta geração (1980 - Atual)

- 2009: Windows 7, trouxe mudanças visuais pequenas em relação ao seu antecessor, mas é mais rápido, estável e fácil de utilizar. SO mais utilizado do mercado.





## Quarta geração (1980 - Atual)

- 2012: Windows 8 e 8.1, tentativa mais radical da Microsoft de alterar o visual do seu SO. A mudança foi motivada por causa da chegada dos dispositivos que respondem ao toque. A mudança na aparência não agradou a todos, o que culminou no “**fracasso**”.





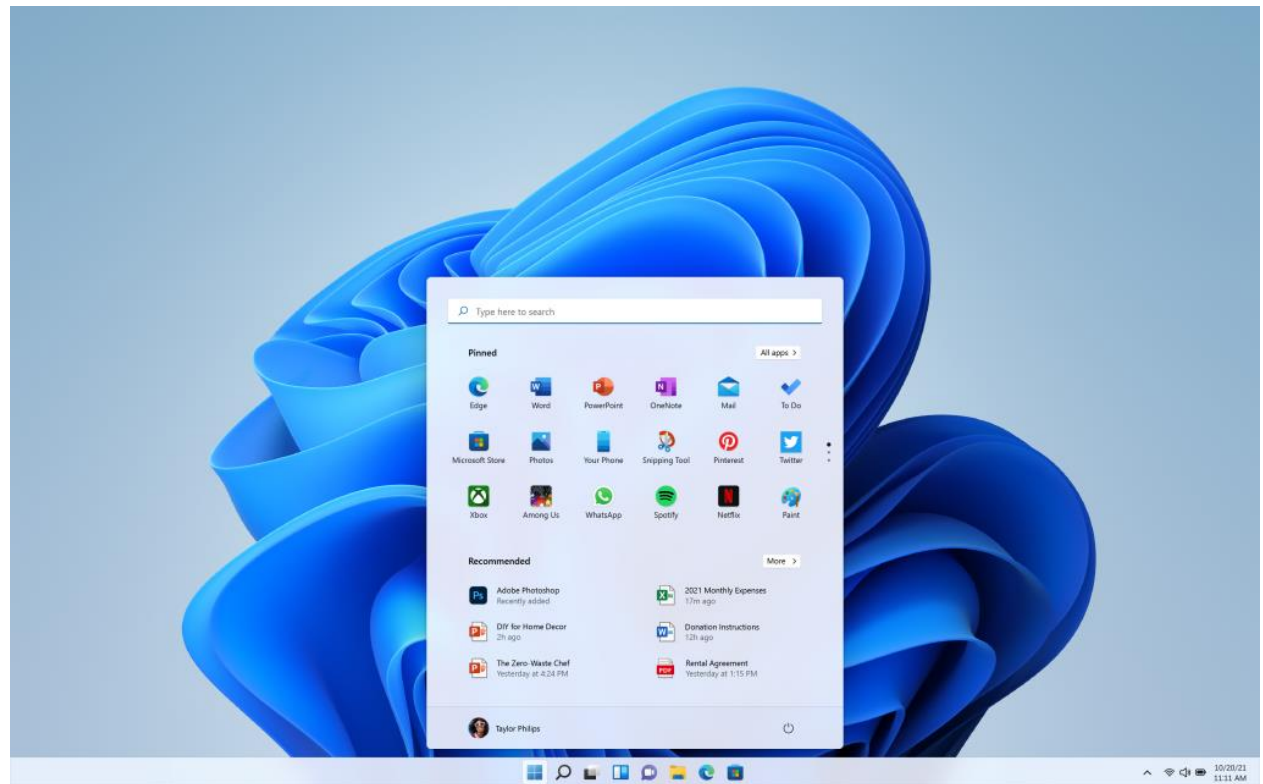
## Quarta geração (1980 - Atual)

- 2015: Windows 10, o principal destaque desta versão foi a reversão da interface para o tradicional paradigma desktop do Windows 7.



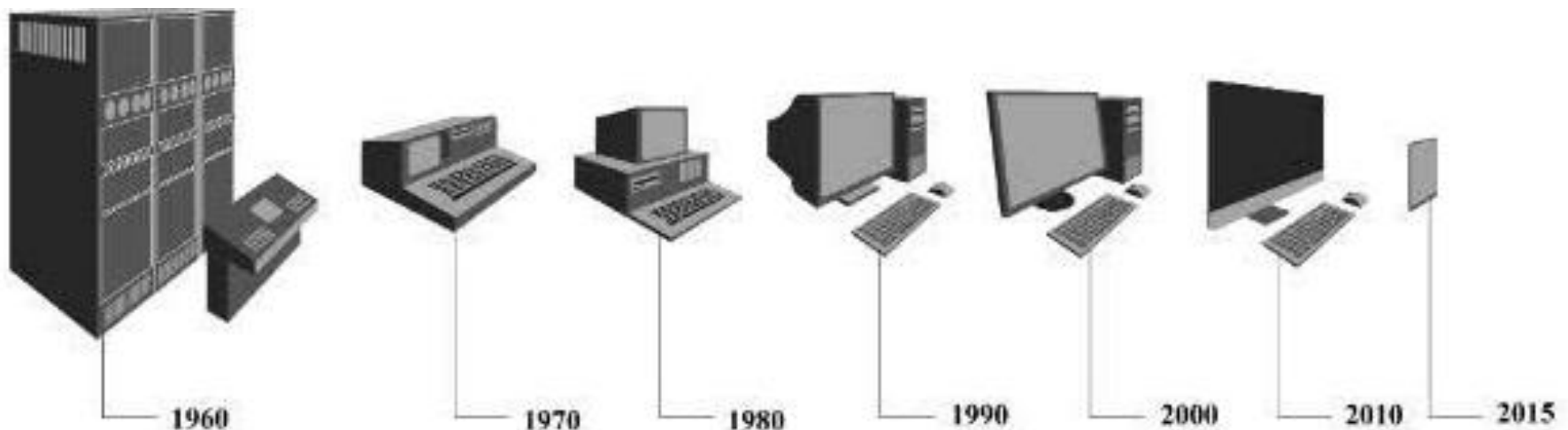
## Quarta geração (1980 - Atual)

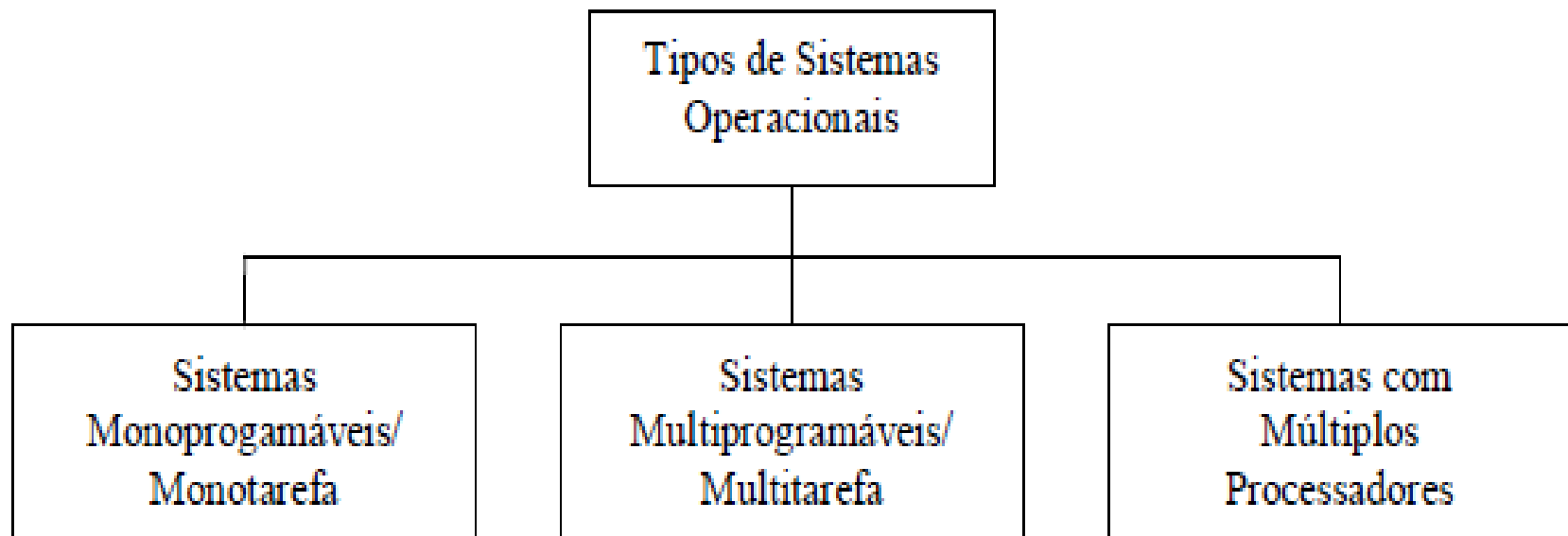
- 2021: Windows 11, sua aceitação foi positiva, com os críticos elogiando o novo design e recursos de produtividade. No entanto, a Microsoft foi criticada por criar confusão sobre os requisitos mínimos.



# Tipos de Sistemas Operacionais

- Tipos de SO e sua evolução estão intimamente relacionados com a evolução do hardware e das aplicações por ele suportadas.
- A evolução dos sistemas operacionais para computadores pessoais e estações de trabalho popularizou vários conceitos e técnicas, antes só conhecidos em ambientes de grande porte.
- Surgiram novos termos para conceitos já conhecidos, que foram apenas adaptados para uma nova realidade.

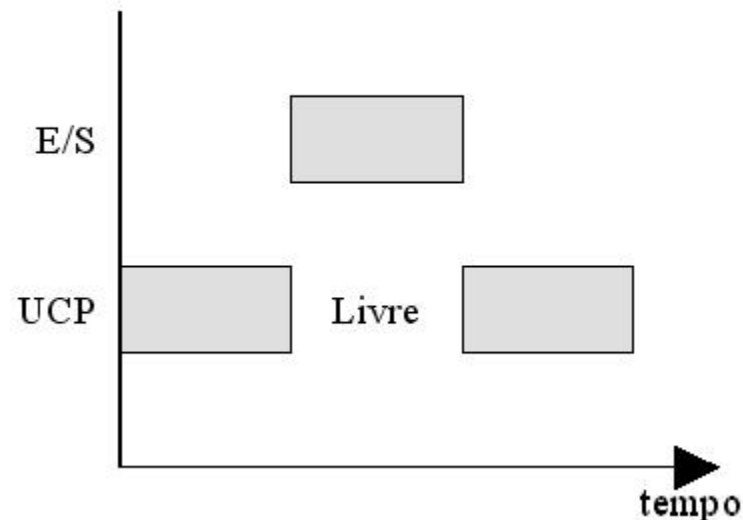




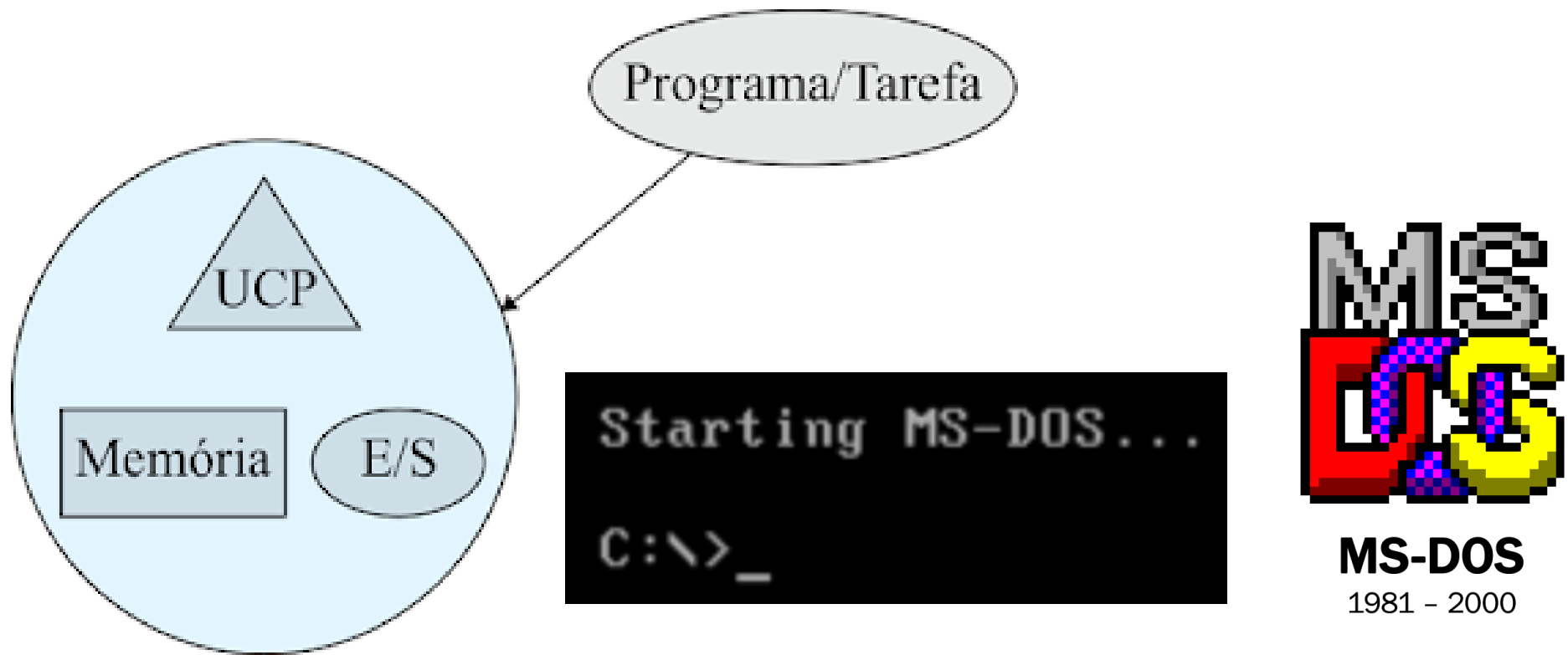
**Figura 13 - Tipos de sistemas operacionais**

## S.Os Monoprogramáveis ou Monotarefas

- 1) Caracterizam-se por permitir que o processador a memória e os periféricos permaneçam **exclusivamente dedicados à execução de um único programa.**
- 2) Aplicações tem controle total do sistema, grande tempo de ociosidade.
- 3) Recursos são **mal** utilizados, entretanto, são implementados com facilidade. Ex: MS-DOS.
- 4) Simples de implementação por não haver preocupação com problemas de compartilhamento de recursos.



## ➤ Sistemas Monoprogramáveis / Monotarefa:





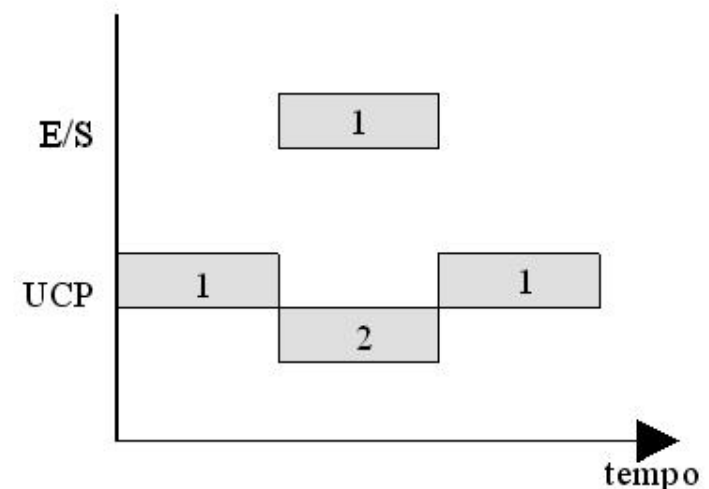
## S.O Multiprogramáveis ou Multitarefas

Os primeiros computadores permitiam que apenas um programa fosse executado de cada vez. Esse programa tinha controle total sobre o sistema e acesso a todos os seus recursos. Por outro lado, os sistemas de computação contemporâneos permitem que vários programas sejam carregados na memória e executados concorrentemente.



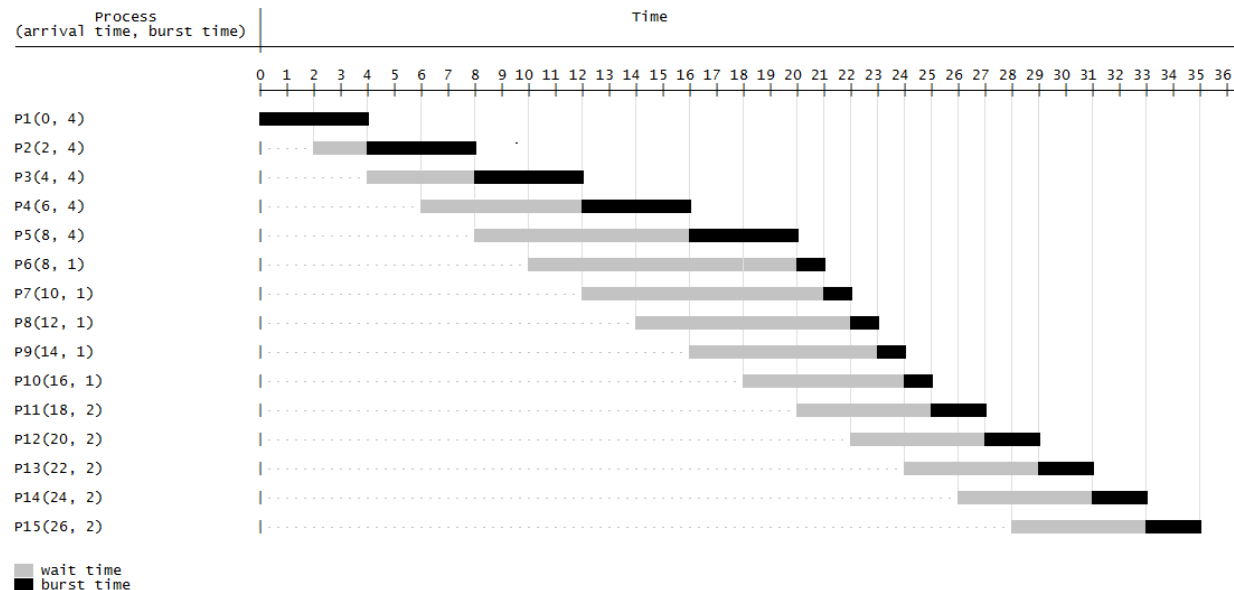
## S.O Multiprogramáveis ou Multitarefas

- 1) A ideia é manter **vários programas** na memória ao mesmo tempo.
- 2) Mais eficientes mas de implementação mais complexa.
- 3) Utilização de Recursos de forma **concorrente**.
- 4) Várias tarefas simultâneas em um único processador: **enquanto uma espera a outra roda**, demandando mecanismos de **trocas rápidas entre os processos**. Ex: Windows, MacOS...

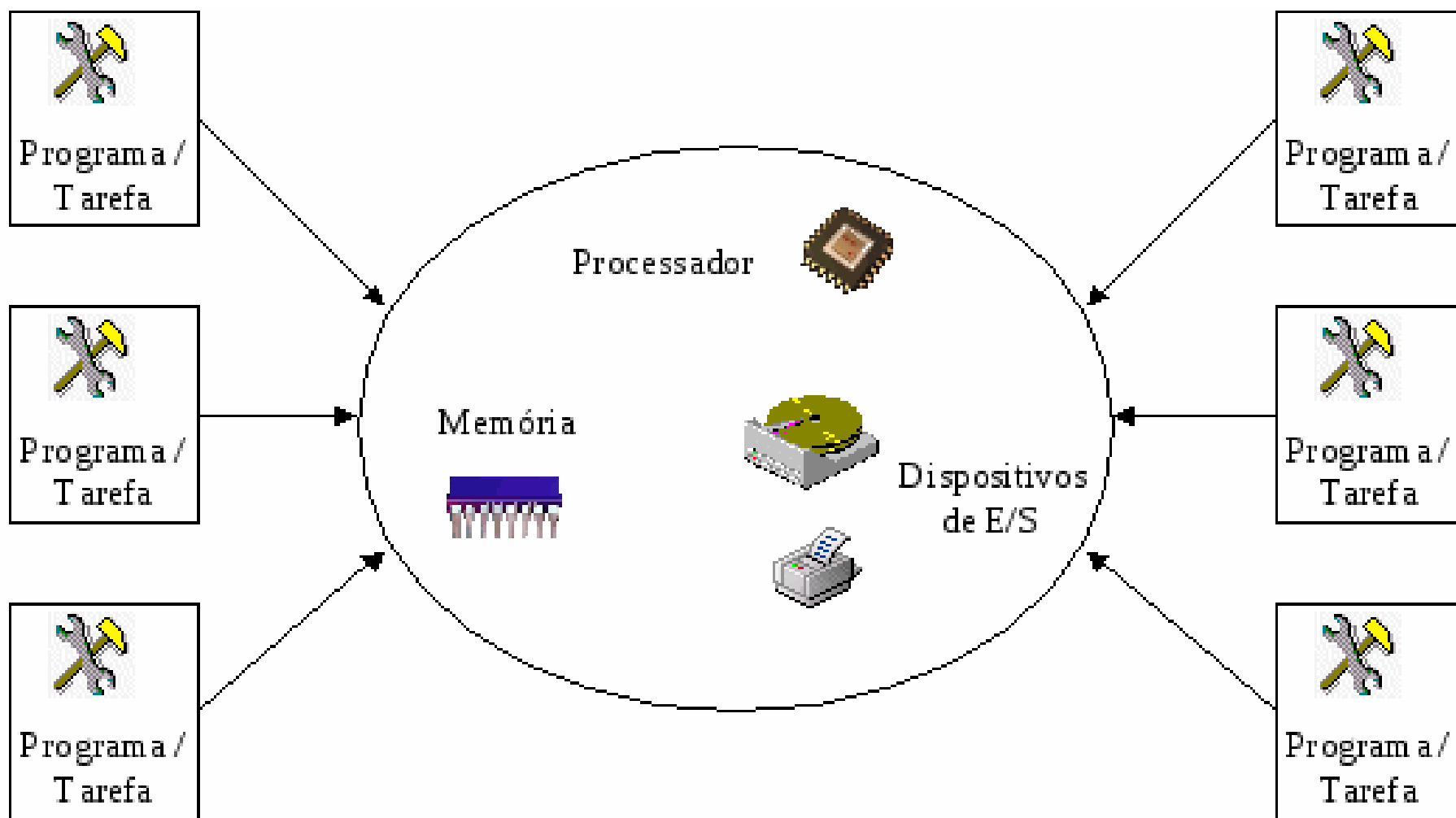


## Pseudo-Paralelismo / Concorrência

Para possibilitar essa "experiência multitarefa", o que ocorre é que a CPU é compartilhada entre os processos **que estão na memória aguardando pela CPU**. Funciona da seguinte forma: **A CPU alterna na execução desses processos** fazendo parecer que os vários processos executam juntos, mas o que houve foi um compartilhamento do tempo de CPU. Essa técnica é conhecida como **time-sharing**.

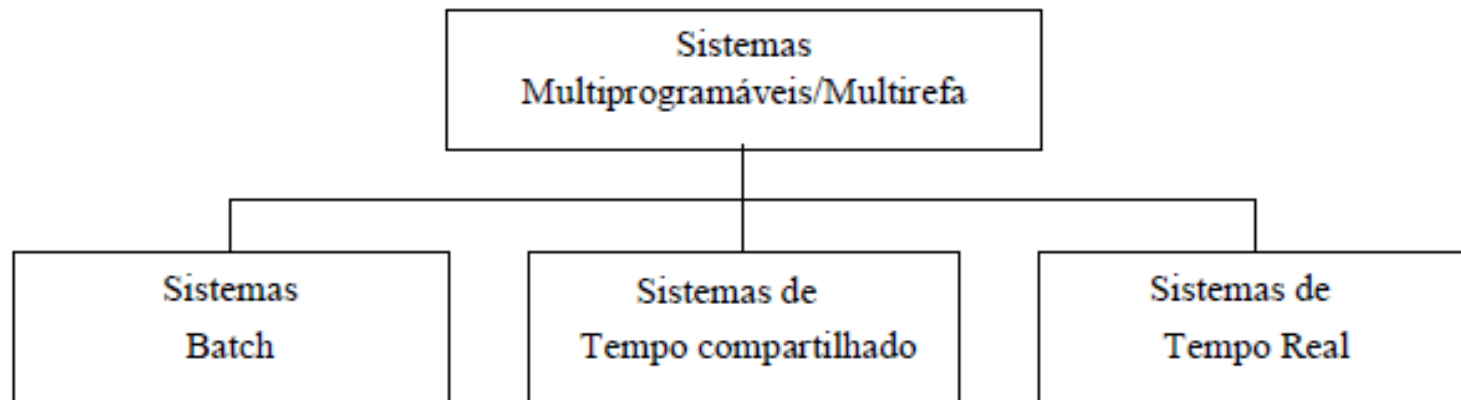


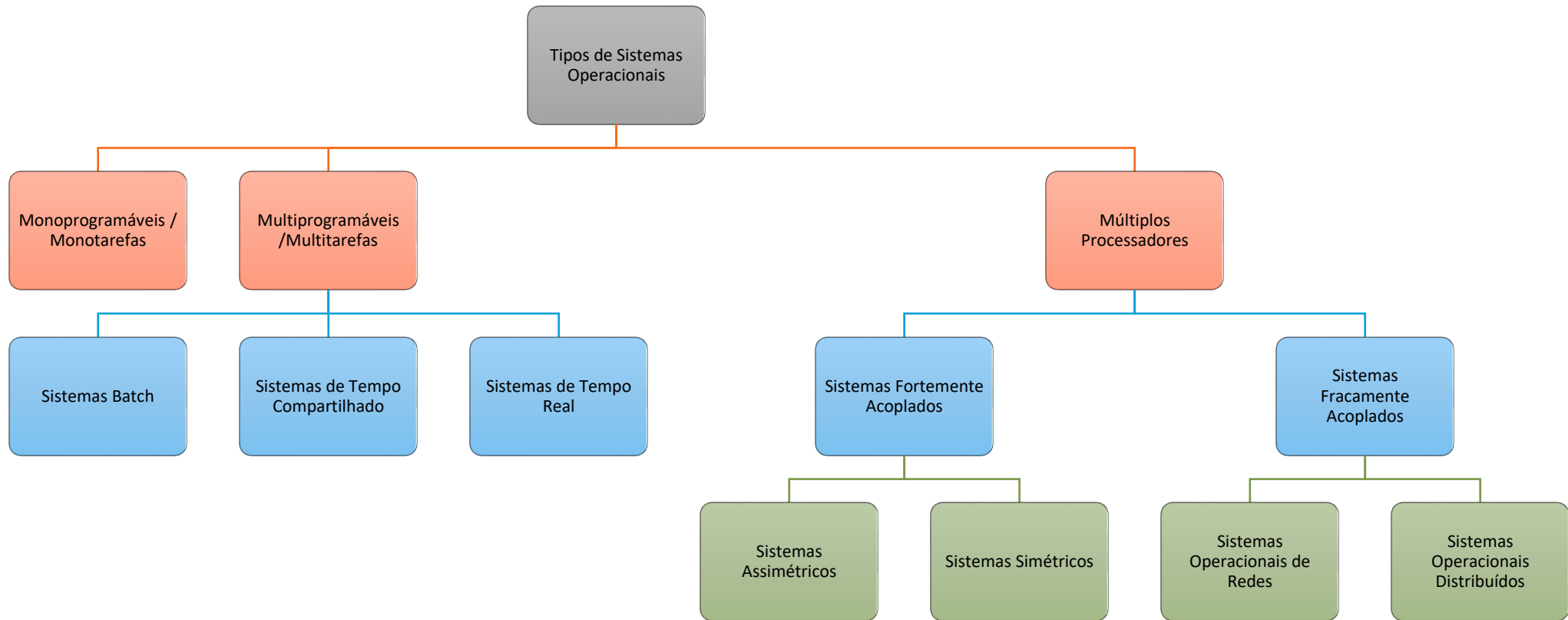
## ➤ Sistemas Multiprogramáveis / Multitarefa:



© Os sistemas multiprogramáveis/multitarefa podem ser classificados pela forma com que suas aplicações são gerenciadas, podendo ser divididos em:

- Sistemas Batch.
- Sistemas de Tempo Compartilhado.
- Sistemas de Tempo Real.







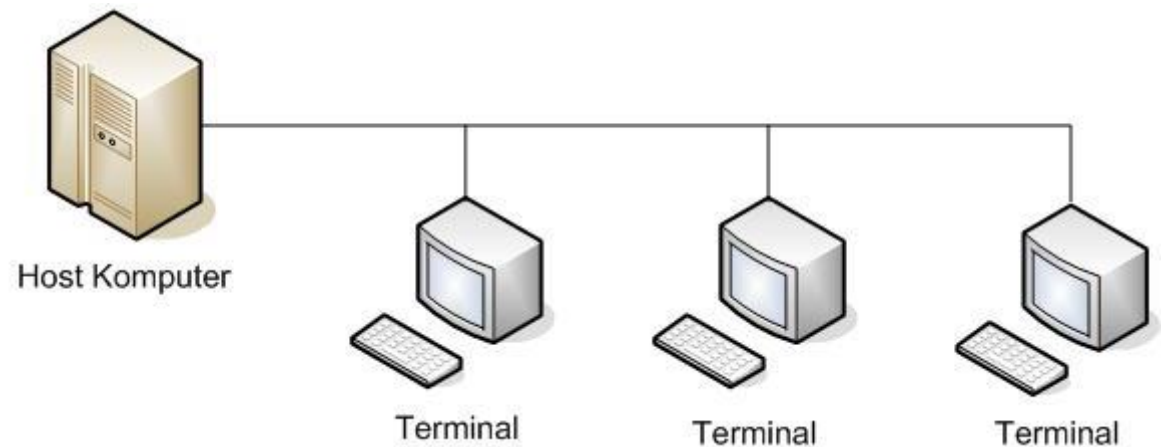
## Sistemas Batch:

- Caracterizam-se por terem seus programas, quando submetidos, armazenados em disco ou fita, onde esperam para ser executados sequencialmente. Também chamados de **jobs**, não exigem interação com os usuários lendo e gravando dados em discos e fitas.
- Exemplo: **backups, videocassete** e todas aquelas onde não é necessária a interação com o usuário.



## Sistemas de Tempo Compartilhado: Time-sharing.

- Permitem a interação dos usuários com o sistema, basicamente através de terminais que incluem vídeo, teclado e mouse.
- O usuário pode interagir diretamente com o sistema em cada fase do desenvolvimento de suas aplicações e se preciso, modificá-las imediatamente.
- Conhecidos como sistemas on-line.

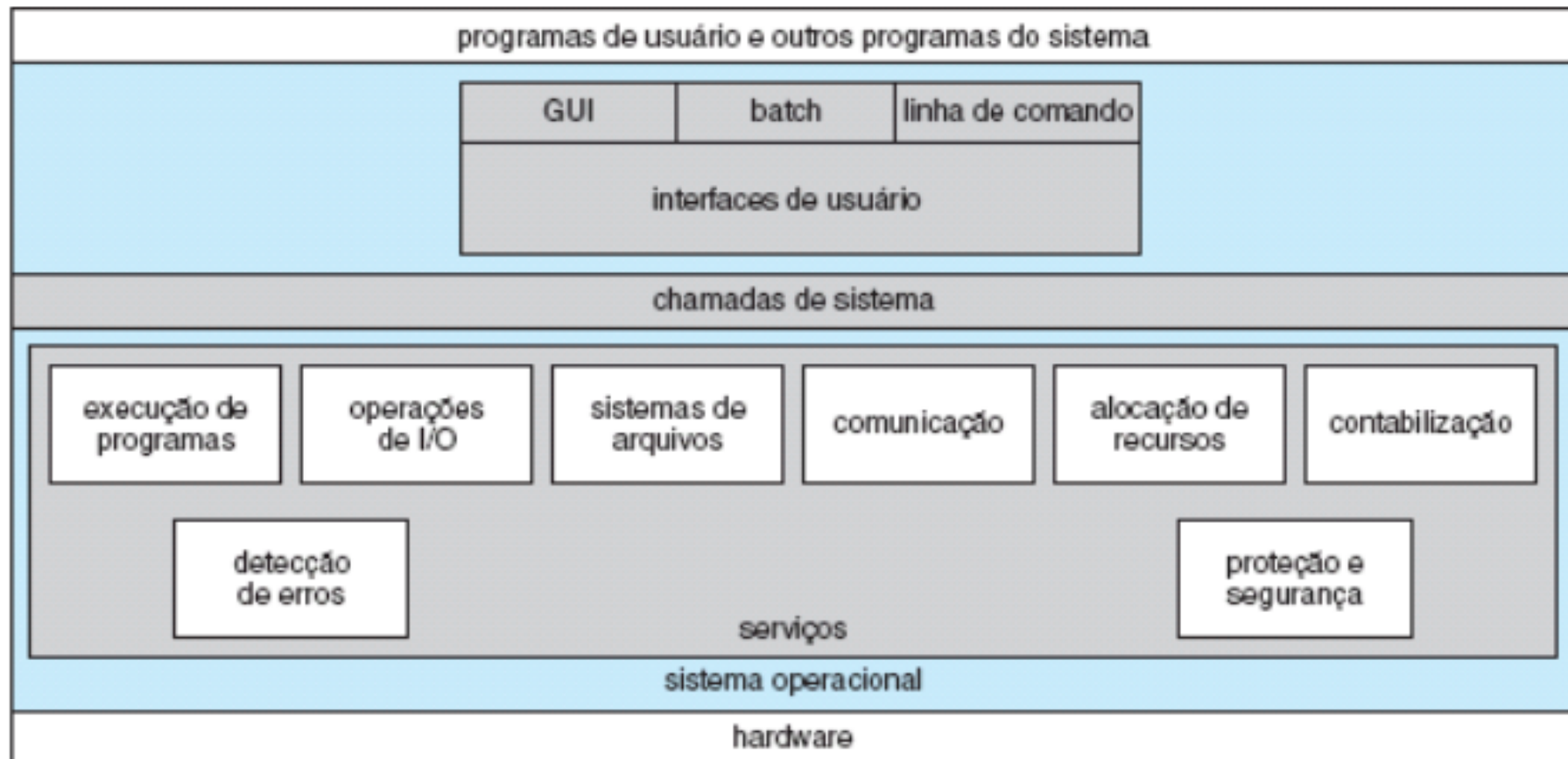


## ⦿ Sistemas de Tempo Real:

- Enquanto em sistemas de tempo compartilhado o tempo de resposta pode variar sem comprometer as aplicações em execução, nos sistemas de tempo real os tempos de resposta devem estar dentro de limites rígidos, que devem ser obedecidos, caso contrário poderão ocorrer problemas irreparáveis.
- **Exemplo:** monitoramento de refinarias de petróleo, controle de tráfego aéreo, de usinas termelétricas e nucleares, ou em qualquer aplicação onde o tempo de resposta é fator fundamental.



## Serviços do Sistema Operacional



## Serviços do Sistema Operacional

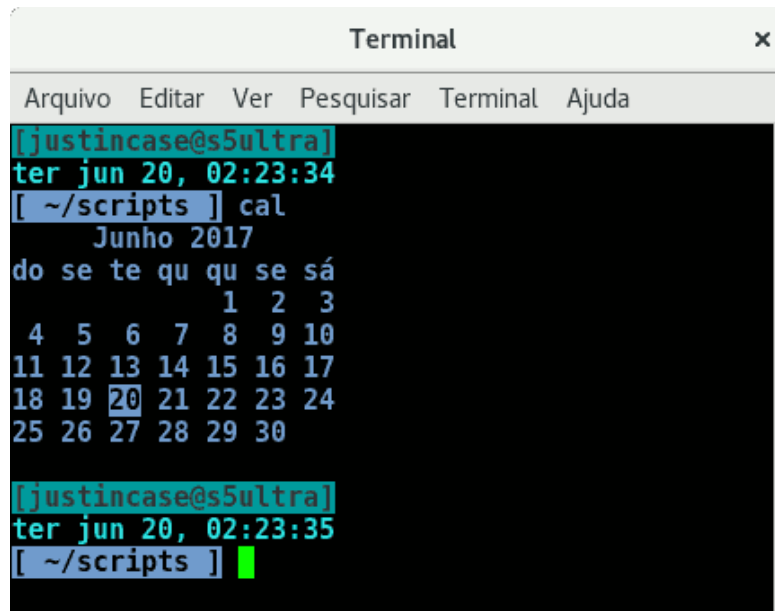
- **Interface de Usuário:** (CLI e GUI);
- **Execução de Programas:** Carregar, executar e finalizar normal ou anormalmente um programa;
- **Operações de I/O:** Requisições de I/O. Exemplos: CD-ROM, impressora, webcam.
- **Manipulação do Sistema de arquivos:** Ler, gravar, excluir arquivos e diretórios, permissões.
- **Comunicações:** Troca de mensagens entre processos. Área de memória compartilhada.
- **Deteção de erros:** falha de conexão, gravação no disco, código do erro para tratamento.
- **Alocação de Recursos:** Múltiplos usuários ativos, rotinas de escalonamento, registradores.
- **Contabilização:** Cobranças por uso, estatísticas, auditoria.
- **Proteção e Segurança:** Controle dos recursos, autenticação, políticas, ferramentas.





## Interface CLI

Em sistemas em que é possível escolher entre vários interpretadores de comandos, esses interpretadores são conhecidos como **shells**. Em sistemas Linux, um usuário pode escolher entre vários shells diferentes. A principal função do interpretador de comandos é capturar e executar o próximo comando especificado pelo usuário. Muitos dos comandos fornecidos, nesse nível, **manipulam arquivos**: criar, excluir, listar, imprimir, copiar, executar, e assim por diante.



```
Terminal
Arquivo Editar Ver Pesquisar Terminal Ajuda
[justincase@s5ultra]
ter jun 20, 02:23:34
[ ~/scripts ] cal
      Junho 2017
do se te qu qu se sá
                1 2 3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30

[justincase@s5ultra]
ter jun 20, 02:23:35
[ ~/scripts ]
```

## Interface CLI

No Linux por exemplo o interpretador de comando não entende o comando ele simplesmente usa o comando para identificar um arquivo a ser carregado na memória e executado. Portanto o comando excluir um arquivo: **rm file.txt**

Busca o arquivo rm, carrega o arquivo na memória e o executa com parâmetro file.txt. A função associada ao comando rm seria totalmente definida pelo código existente no arquivo rm. Dessa forma os programadores podem adicionar facilmente novos comandos ao sistema, criando arquivos novos arquivos com nomes apropriados, deixando o interpretador de comandos pequeno, leve e sem precisar atualizar a cada comando.

## Interface GUI

Uma segunda estratégia de comunicação com o sistema operacional é por meio de uma interface gráfica de usuário amigável ou GUI. Aqui, em vez de dar entrada em comandos, diretamente, por meio de uma interface de linha de comando, os usuários empregam um sistema de janelas e menus baseado em mouse, caracterizado por uma simulação de área de trabalho.



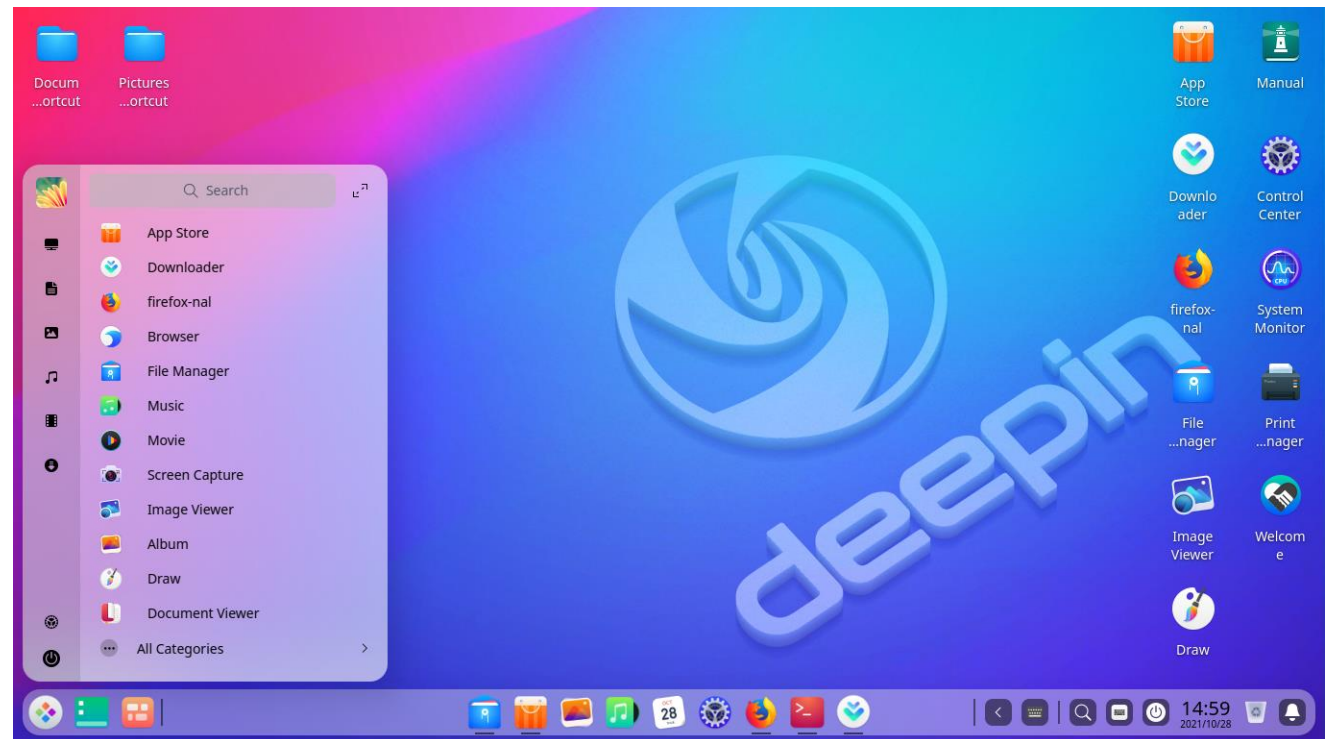
## Interface GUI

As interfaces gráficas de usuário surgiram, em parte, por causa de pesquisas que ocorreram no início dos anos 1970, no centro de **pesquisas Xerox PARC**. A primeira GUI surgiu no computador **Xerox Alto, em 1973**. No entanto, as interfaces gráficas disseminaram-se nos anos 1980, com o advento dos computadores Apple Macintosh.



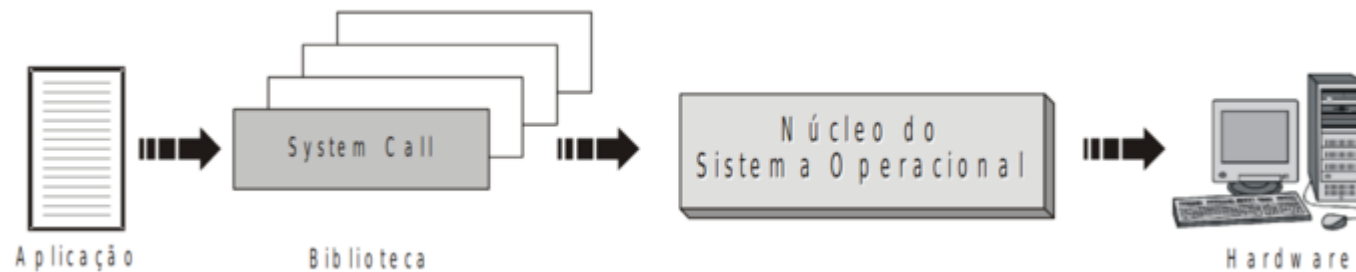
## Interface GUI

Várias GUIs estão disponíveis, no entanto nem todas utilizam código fonte aberto. As mais famosas interfaces para Linux são: Gnome, KDE, Cinnamon, Mate, XFCE, Pantheon (Elementary), DDE (Deepin), Unity.



## Chamadas de Sistemas

Se uma instrução precisa realizar alguma instrução privilegiada (**imprimir um arquivo**), ela realiza uma chamada de sistema, que altera do modo usuário para o modo kernel. Ex: ler um arquivo (open), imprimir (printf), criar um processo (fork), etc.

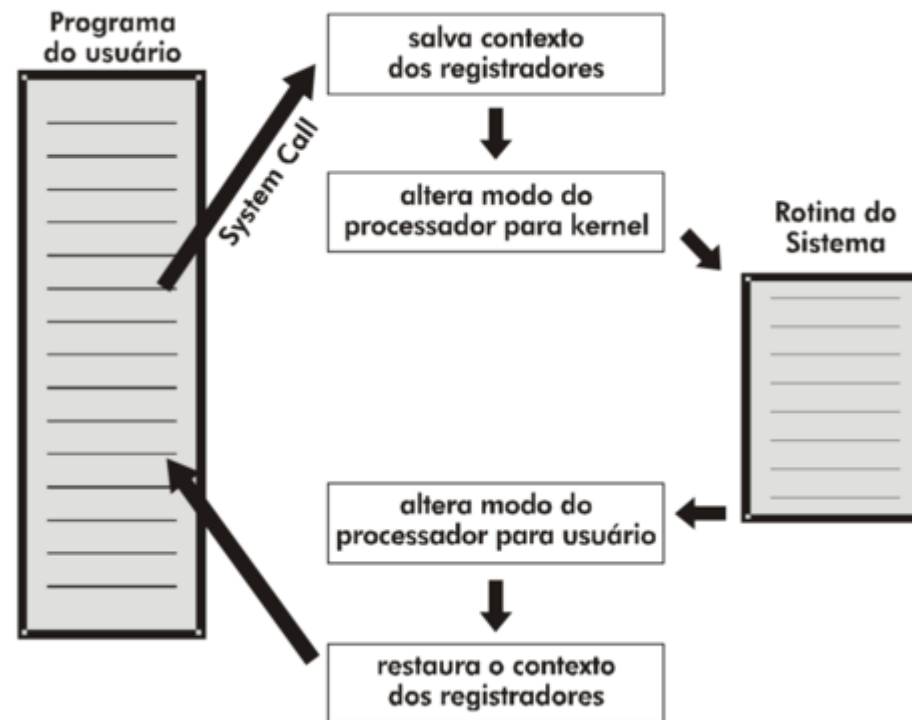




## Chamadas de Sistema

As chamadas de sistema fornecem uma interface com os serviços disponibilizados por um sistema operacional. Geralmente, essas chamadas estão disponíveis como rotinas escritas em C e C++. Exemplo: ler dados em um arquivo e copiá-los em outro arquivo. A primeira entrada de que o programa precisará são os nomes dos dois arquivos. o arquivo de entrada e o arquivo de saída. Uma vez que os dois nomes de arquivo tenham sido obtidos, o programa deve abrir o arquivo de entrada e criar o arquivo de saída. Cada uma dessas operações requer outra chamada de sistema. Condições de erro que podem ocorrer, para cada operação, podem requerer chamadas de sistema adicionais.

## Passos para chamada de Sistema



## Chamadas de Sistema

Quando o programa tentar abrir o arquivo de entrada, por exemplo, pode descobrir que não há arquivo com esse nome, ou que o arquivo está protegido contra acesso. Nesses casos, o programa deve exibir uma mensagem no console (outra sequência de chamadas de sistema) e, então, terminar anormalmente (outra chamada de sistema). Se o arquivo de entrada existe, devemos criar um novo arquivo de saída.

```
rohith@rohith-pc: /home
rohith@rohith-pc:~$ cd ..
rohith@rohith-pc:/home$ ls -ld shadi
drwxr-xrwx 2 root root 4096 Jul  1 17:35 shadi
rohith@rohith-pc:/home$ rm -r shadi
rm: cannot remove 'shadi': Permission denied
rohith@rohith-pc:/home$
```

## Chamadas de Sistema

Podemos descobrir que já existe um arquivo de saída com o mesmo nome. Essa situação pode fazer com que o programa aborte (uma chamada de sistema), ou podemos excluir o arquivo existente (outra chamada de sistema) e criar um novo (mais uma chamada de sistema). Outra opção, em um sistema interativo, é perguntar ao usuário (por meio de uma sequência de chamadas de sistema para exibir a mensagem de alerta e para ler a resposta a partir do terminal) se deseja substituir o arquivo existente ou abortar o programa.

```
Command Prompt

C:\Users\wikihow\Documents>cd C:\Users\wikiHow\Documents\Myfiles

C:\Users\wikihow\Documents\Myfiles>copy presentation.pptx E:\backup\
1 file(s) copied.

C:\Users\wikihow\Documents\Myfiles>xcopy

C:\Users\wikihow\Documents\Myfiles>xcopy C:\Users\wikiHow\Documents\Myfiles
C:\Users\wikiHow\Documents\Myfiles\Company Letter.docx
C:\Users\wikiHow\Documents\Myfiles\Monthly company budget1.xlsx
C:\Users\wikiHow\Documents\Myfiles\My Story.docx
Overwrite E:\Backup\Presentation.pptx (Yes/No/All)? y
C:\Users\wikiHow\Documents\Myfiles\Presentation.pptx
4 File(s) copied

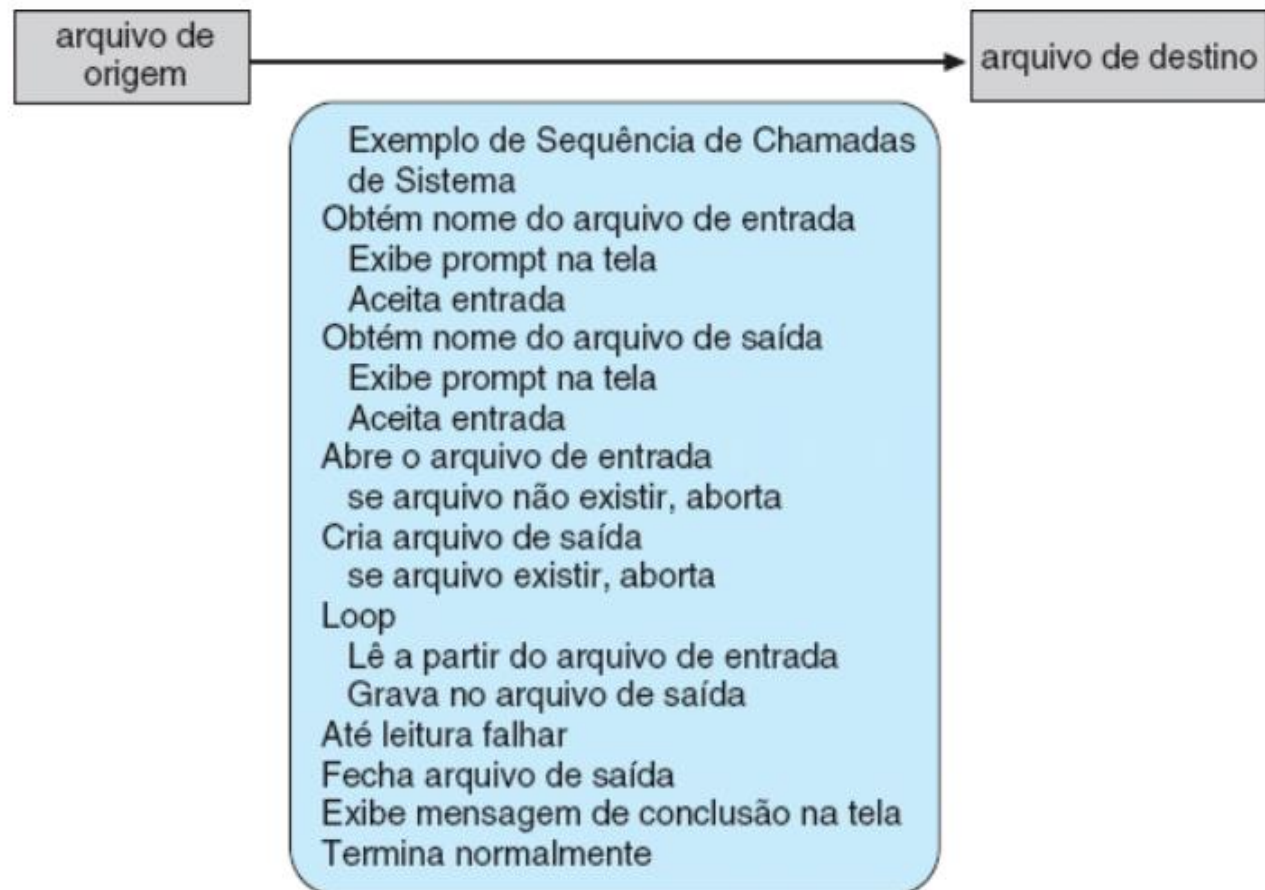
C:\Users\wikihow\Documents\Myfiles>
```

## Chamadas de Sistema

Quando os dois arquivos estão definidos, entramos em um loop que lê o arquivo de entrada (uma chamada de sistema) e grava no arquivo de saída (outra chamada de sistema). Cada operação de leitura e gravação deve retornar informações de status referentes a várias condições de erro possíveis

Na entrada, o programa pode entender que o fim do arquivo foi alcançado ou que houve uma falha de hardware na leitura (como um erro de paridade). A operação de gravação pode encontrar vários erros, dependendo do dispositivo de saída (por exemplo, não há mais espaço em disco). Para concluir, após o arquivo inteiro ser copiado, o programa pode fechar os dois arquivos (outra chamada de sistema), exibir uma mensagem no console ou janela (mais chamadas de sistema) e, por fim, terminar normalmente (a última chamada de sistema).

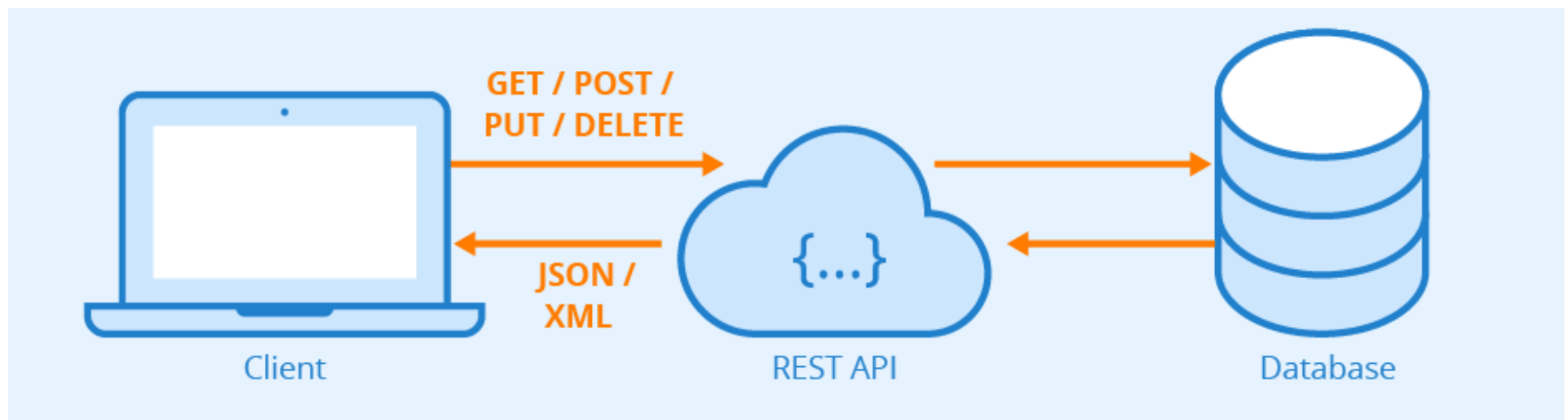
## Chamadas de Sistema





## Chamadas de Sistema

Frequentemente, os sistemas executam milhares de chamadas de sistema por segundo. No entanto, a maioria dos programadores nunca vê esse nível de detalhe. Normalmente, os desenvolvedores de aplicações projetam programas de acordo com uma interface de programação de aplicações (API — application programming interface).

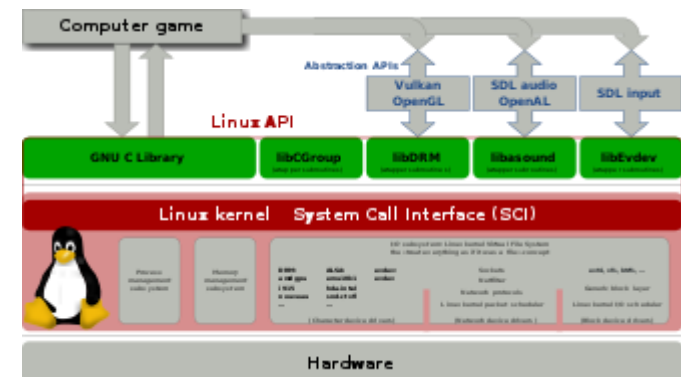


## Chamadas de Sistema

A API especifica um conjunto de funções que estão disponíveis para um programador de aplicações, incluindo os parâmetros que são passados a cada função e os valores de retorno que o programador pode esperar. As três APIs mais comuns, disponíveis para programadores de aplicações, são a **API Windows** para sistemas Windows, a **API POSIX** para sistemas baseados em POSIX (que incluem virtualmente todas as versões do UNIX, Linux e Mac OS X) e a **API Java** para programas que são executados na máquina virtual Java.

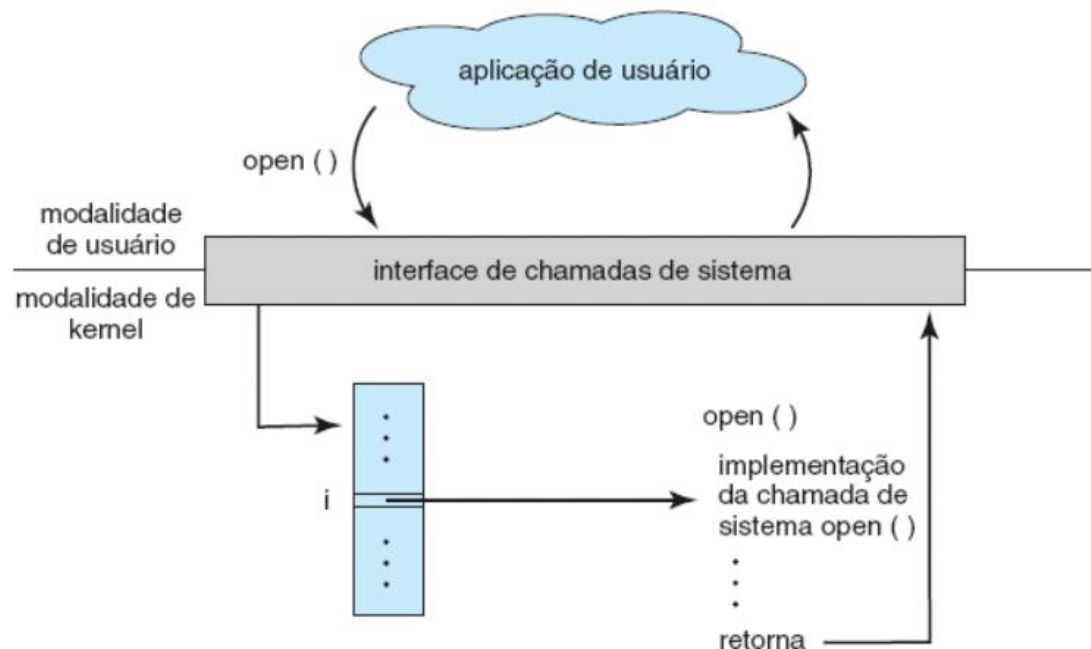
## Chamadas de Sistema

Um programador acessa uma API por meio de uma biblioteca de códigos fornecida pelo sistema operacional. No caso do UNIX e do Linux, para programas escritos na linguagem C, a biblioteca se chama **libc**. Cada sistema operacional tem seu próprio nome para cada chamada de sistema. Em segundo plano, as funções que compõem uma API invocam tipicamente as chamadas de sistema reais em nome do programador de aplicações.



## Chamadas de Sistema

Por exemplo, a função `CreateProcess ( )` do Windows (que, obviamente, é usada para criar um novo processo) na verdade invoca a chamada de sistema `NTCreateProcess ( )` no kernel do Windows.

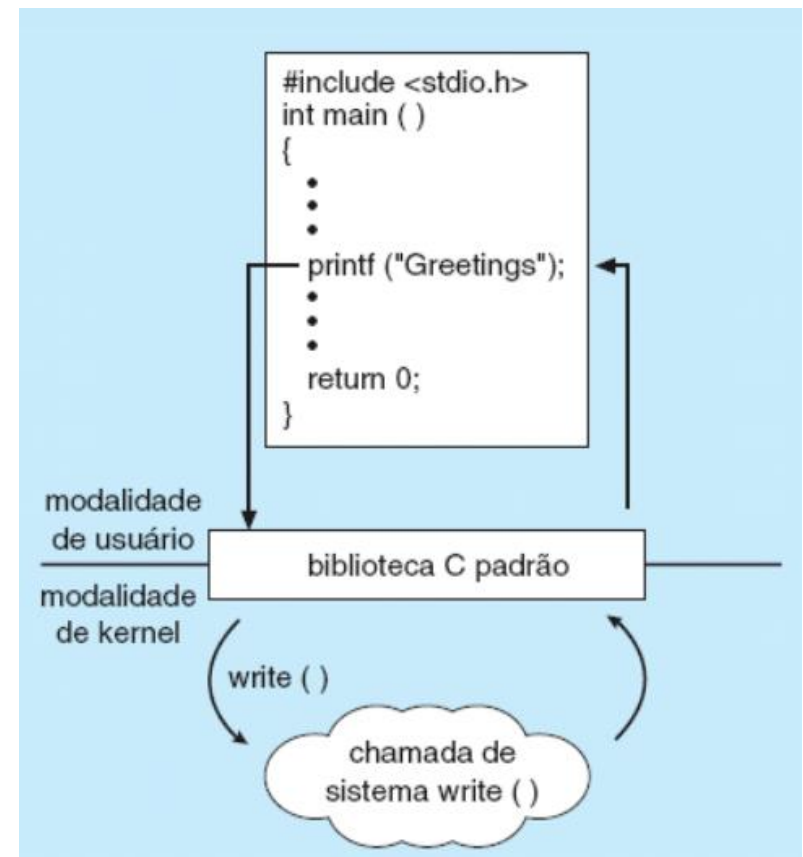


## Exemplos de Chamadas de Sistema

	Windows	Unix
<b>Controle de Processos</b>	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
<b>Manipulação de Arquivos</b>	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
<b>Manipulação de Dispositivos</b>	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
<b>Manutenção de Informações</b>	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
<b>Comunicações</b>	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shm_open() mmap()
<b>Proteção</b>	SetFileSecurity() InitializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

## Chamadas de Sistema

A biblioteca C padrão fornece parte da interface de chamadas de sistema de muitas versões do UNIX e Linux. Como exemplo, **suponha que um programa em C invoque o comando `printf ( )`**. A biblioteca C intercepta essa chamada e invoca a chamada (ou chamadas) de sistema necessária no sistema operacional. Nesse exemplo, **a chamada de sistema `write ( )`**. A biblioteca C toma o valor retornado por `write ( )` e o passa de volta ,o programa do usuário.





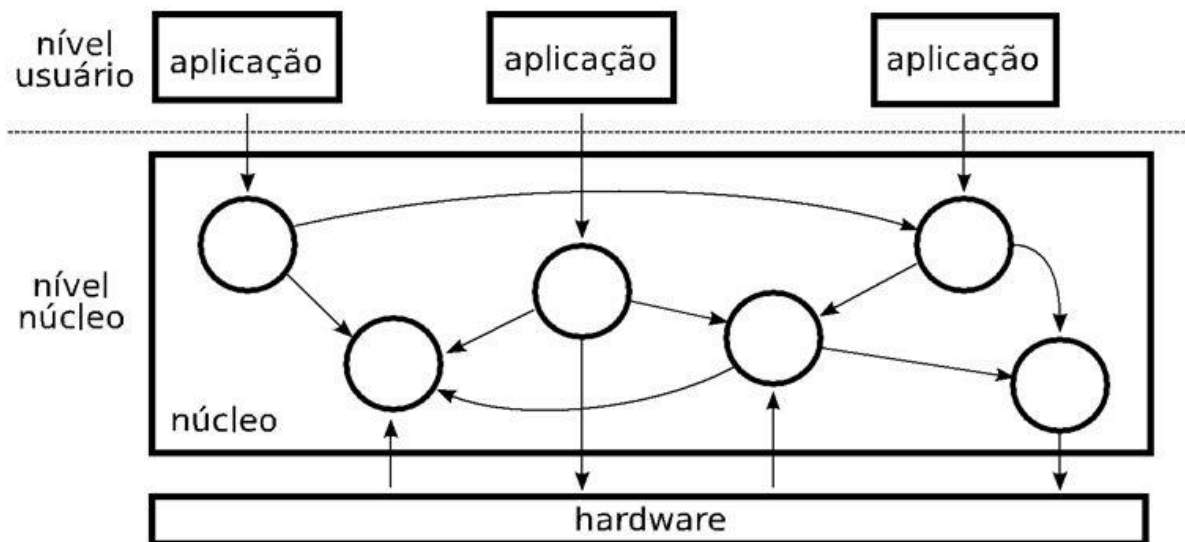
## Estrutura do S.O

SOs são normalmente grandes e complexas coleções de rotinas de software. Projetistas devem dar grande ênfase à sua organização interna e estrutura.

- 1) Monolítico
- 2) Microkernel
- 3) Kernel Híbrido
- 4) Camadas
- 5) Máquina Virtual

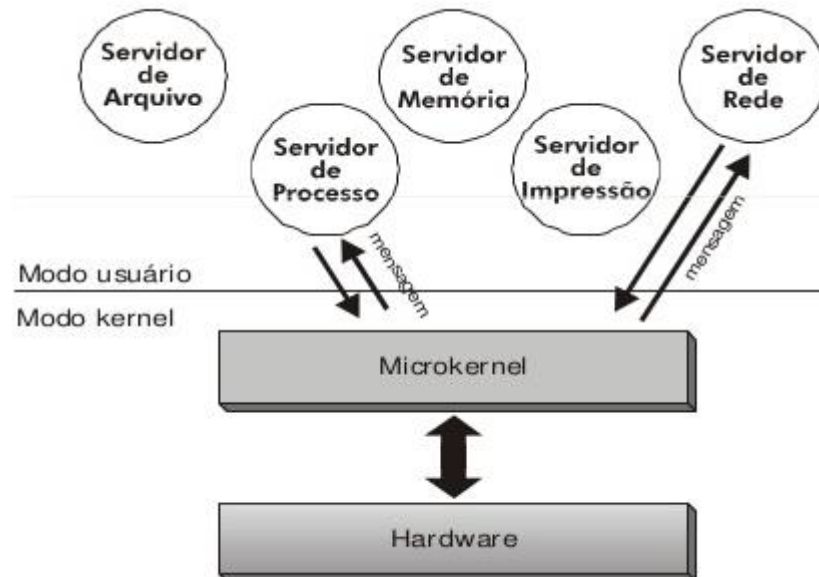
## Monolítico

S.O é um único módulo, consiste de um conjunto de programas que executam sobre o hardware, como se fosse um único programa. Os programas dos usuários invocam rotinas do S.O. Ex: **Linux, Android, Windows, Unix, MS-DOS.**



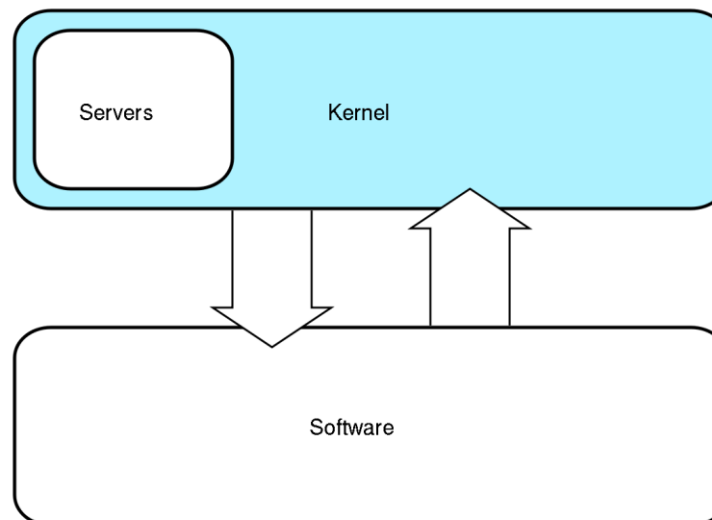
## Microkernel

Busca tornar o núcleo do S.O o menor possível, sendo a principal função do núcleo, gerenciar a comunicação entre esses processos. Apenas o Kernel responsável pela comunicação entre clientes e servidores, executa no modo kernel. Exemplos: **Hurd, Minix, Symbian.**



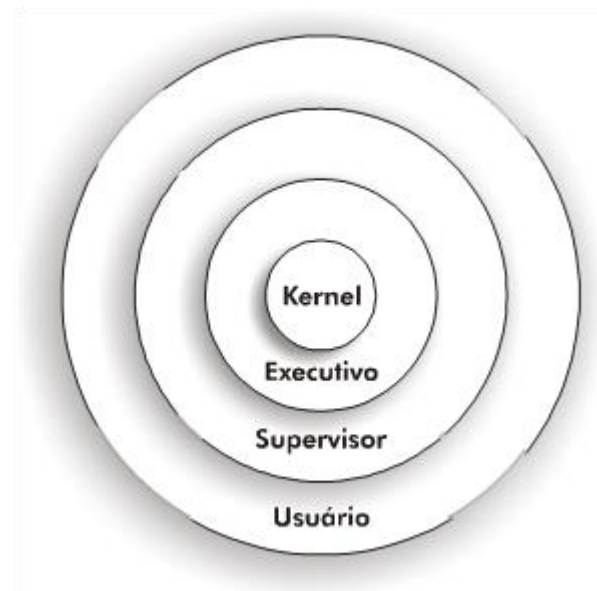
## Kernel Híbrido

Kernel híbrido são um acordo entre o desenvolvimento de microkernel e kernel monolíticos. Isto implica executar alguns serviços (como a pilha de rede ou o sistema de arquivos por exemplo) no espaço do núcleo para reduzir o impacto na performance. Exemplos: **BeOS, MacOS X. Windows NT, Windows Server, etc.**



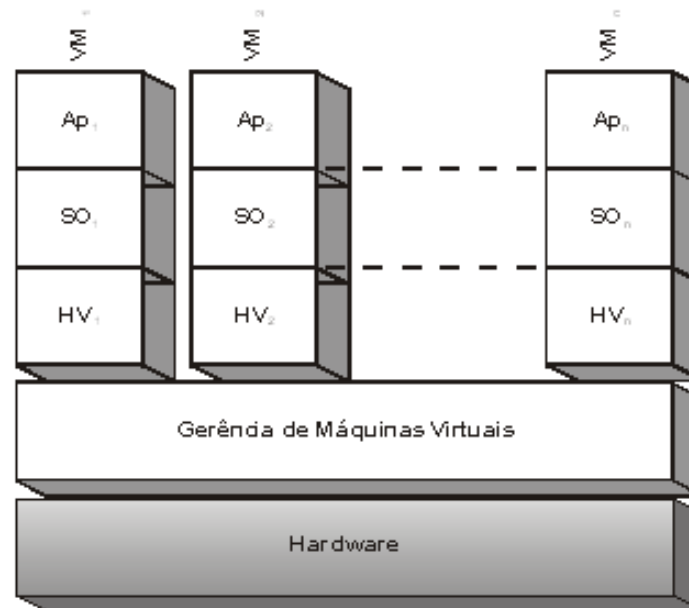
## Camadas

A ideia é criar um sistema modular e hierárquico. Na estrutura de camadas o sistema é dividido em níveis sobrepostos, cada camada oferece um conjunto de funções que podem ser utilizadas pelas camadas superiores. Exemplo: **OpenVMS e Multics**.



## Arquitetura de Máquina Virtual

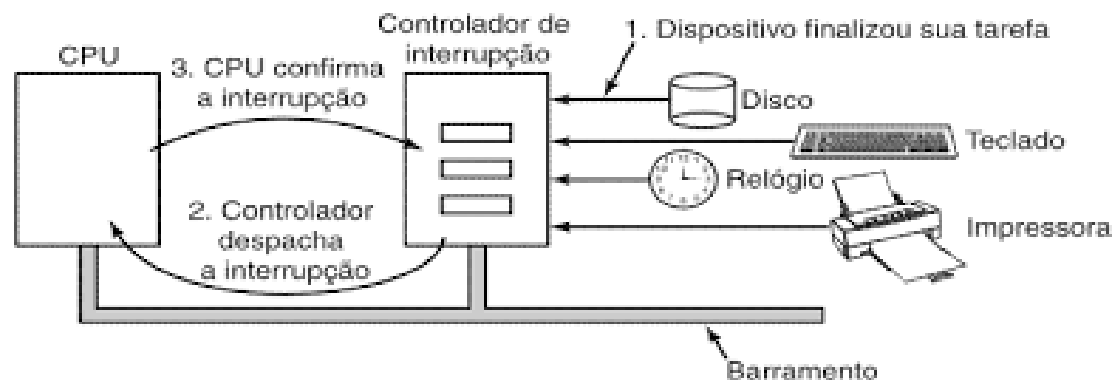
O Modelo de Máquina Virtual (VM) cria um nível intermediário entre o hardware e o sistema operacional, denominado gerenciador de máquinas virtuais. Ex: **IBM360/CMS**.



## Interrupções

Evento externo ao processador, gerado por dispositivos que precisam da atenção do S.O, é uma chamada gerada pelo hardware podendo ser:

- 1) Interrupções de hardware (evento externo);
- 2) Um sinal elétrico no hardware; e
- 3) Dispositivos de E/S ou o clock.







Obrigado!