

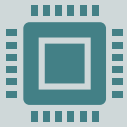
ESTRUTURA DE SISTEMAS OPERACIONAIS

A series of horizontal lines in teal and light blue colors, with varying lengths and offsets, creating a modern, layered effect across the middle of the slide.

- O sistema operacional é formado por um conjunto de rotinas que oferece serviços aos usuários e às suas aplicações.
- Ele disponibiliza uma grande variedade de atividades que nós usuários podemos solicitar para satisfazer nossas necessidades.
- Essa variedade de atividades , que são realizadas pela utilização dos softwares, são executadas utilizando o hardware.



O conjunto de rotinas que o sistema operacional executa é o que chama-se de núcleo do sistema operacional ou kernel.



A função do núcleo do SO ou Kernel é conectar ambos: software e hardware, permitindo assim uma eficiente comunicação.



Contudo os sistemas operacionais são formados por algo além do núcleo.



Os sistemas operacionais possuem também uma linguagem de comandos e diversos utilitários de apoio que são usados para complementar o sistema operacional.



Os usuários e aplicações se comunicam com o núcleo do sistema operacional de maneiras distintas.

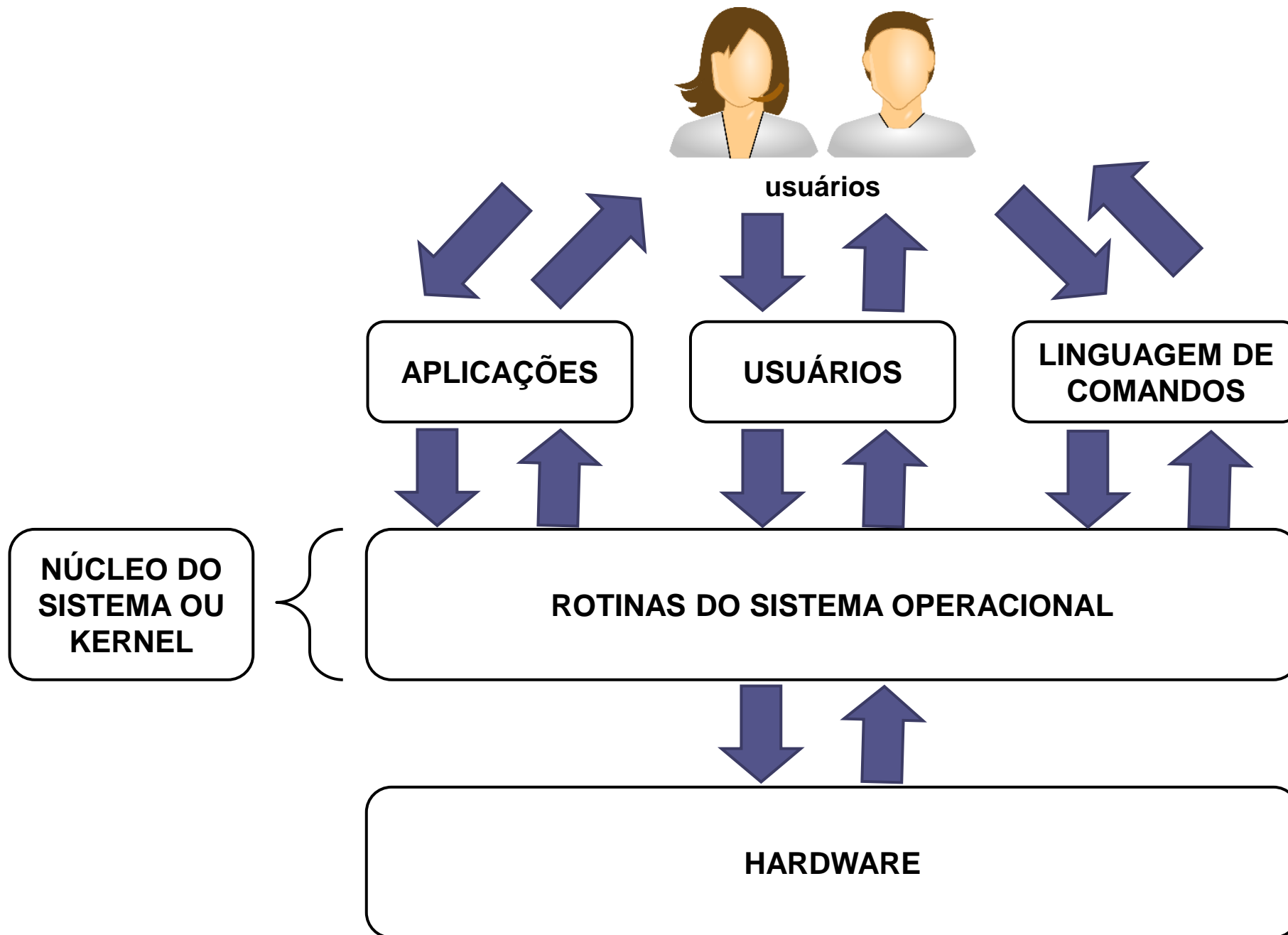


Em geral, os usuários usam aplicações que invocam as rotinas do sistema operacional, por exemplo, quando gravamos um arquivo.

- Além disso, os usuários também usam a linguagem de comandos do sistema operacional para executar tarefas no sistema operacional e também usam programas utilitários para tarefas mais complexas como compilação e transmissão de arquivos por exemplo.

O ponto é que cada sistema operacional possui sua própria linguagem de comandos, seus próprios utilitários e demais componentes que formam sua estrutura.

A estrutura de um sistema operacional pode variar conforme o projeto do próprio sistema operacional mas em geral temos os seguintes componentes:



- A estrutura do núcleo, ou seja, a maneira como o código do sistema é organizado e o inter-relacionamento entre seus diversos componentes podem variar conforme a concepção do SO.

KERNEL - Núcleo do SO



O núcleo sempre gerencia a execução das rotinas (processos).



As rotinas são executadas simultaneamente, sem ordem predefinida, com base em eventos que não ocorrem ao mesmo tempo (assíncronos).



Isso ocorre diferentemente dos sistemas aplicativos, onde as rotinas ocorrem na ordem determinada pelo programa.




Muitas dessas rotinas do SO são gerados por hardware e por tarefas internas do sistema operacional.


- Principais funções do kernel do SO:
 - Tratamento de interrupções e exceções.
 - Criação e eliminação de processos e threads (pequenos programas que trabalham como um subsistema, sendo uma forma de um processo se autodividir em duas ou mais tarefas que se executam simultaneamente).
 - Sincronização e comunicação entre processos e threads.
 - Escalonamento e controle de processos e threads.

- Gerência de memória.
- Gerência do sistema de arquivos.
- Gerência de dispositivos de E/S.
- Suporte a redes locais e distribuídas.
- Contabilização do uso do sistema.
- Auditoria e segurança do sistema.

Com a multiprogramação, é natural que usuários diferentes utilizem o mesmo recurso computacional (memória e processador, por exemplo).



Nesse caso o SO tem que garantir a confiabilidade na execução das tarefas e a integridade do próprio sistema operacional.



Para garantir essa integridade e segurança existem os modos de acesso ao kernel.

Outra função do SO é garantir que nenhum processo monopolize o uso da CPU, permitindo que haja uma alternância dos processos executados concorrentemente.

O SO precisa, ainda, garantir que cada programa tenha sua área de memória protegida de outros programas para que um programa não acesse nenhum endereço de memória que esteja em uso por outro programa.

Caso um programa acesse um endereço de memória em uso por outro programa, deve ser gerado um erro, uma exceção.



O mesmo acontece com o gravação dos dados em disco (gerência do sistema de arquivos).



O SO deve garantir que os dados de cada usuário sejam gravados em disco de forma confiável, considerando que existem vários programas, de vários usuários, gravando dados no disco ao mesmo tempo.



Neste caso é função do sistema operacional garantir a integridade e confiabilidade dos dados armazenados pelos vários programas executados.

O sistema operacional deve permitir ainda que mais de um usuário possa ter acesso simultâneo aos dados de um arquivo.

Todas essas situações fazem com que o SO tenha que implementar uma série de controles e proteção para garantir a execução de várias tarefas e o compartilhamento seguro e confiável dos recursos do computador (disco, memória, impressora, etc.).

Dois desses mecanismos são as interrupções e as exceções.

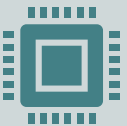
Modos de Acesso ao Kernel



Em todo projeto de SO os projetistas se preocuparam em garantir a integridade do próprio sistema, protegendo o núcleo e o serviços do sistema de acessos indevidos por uma aplicação do usuário.



Caso uma aplicação do usuário tenha acesso ao núcleo e realize alguma operação que altere sua integridade então todo o sistema ficará comprometido.



Muitos SO's utilizam um mecanismo de segurança implementado no hardware do processador e conhecido como modo de acesso.

Normalmente, são dois modos:

- **Modo usuário:** é o que executa as instruções não privilegiadas, tem acesso a um número reduzido de instruções e não oferece risco ao SO;
- **Modo kernel:** é o modo que executa as instruções consideradas privilegiadas. Somente usuários com permissões específicas possuem autorização para manipular no modo kernel.

O modo de acesso é determinado por um conjunto de bits no registrador de status.

O hardware verifica este registrador para determinar se uma instrução pode ou não ser executada.

Interpretador de Comandos

Também conhecido como Shell, o interpretador de comandos interpreta as solicitações feitas pelo usuário.

Alguns SO's incluem o interpretador de comandos no kernel. Outros, como Windows e UNIX, tratam-no como um programa especial que está sendo executado quando uma tarefa é iniciada e o usuário faz a solicitação.

Existem basicamente dois tipos:

Texto: quando as solicitações são feitas por linhas de comando, como o MS-DOS. São comandos específicos e técnicos, parametrizados sob uma rigorosa sintaxe;

```
C:\DOS>chkdsk
Volume Serial Number is 3E76-4B58

2,146,467,840 bytes total disk space
    131,072 bytes in 2 hidden files
    32,768 bytes in 1 directories
    7,405,568 bytes in 124 user files
2,138,898,432 bytes available on disk

    32,768 bytes in each allocation unit
    65,505 total allocation units on disk
    65,274 available allocation units on disk

    655,360 total bytes memory
    602,704 bytes free

Instead of using CHKDSK, try using SCANDISK.  SCANDISK can reliably detect
and fix a much wider range of disk problems.  For more information,
type HELP SCANDISK from the command prompt.

C:\DOS>_
```

Gráfico

A interface gráfica surgiu na década de 1970, por meio da Xerox. A primeira versão do Windows era baseada em uma interface gráfica, para o sistema operacional MS-DOS.



Esse tipo de interpretador de comandos se caracteriza pelas janelas e ícones. O usuário faz sua solicitação utilizando normalmente o mouse. Atualmente, temos a possibilidade de fazer as solicitações via “touch screen”, ou seja, com toques na tela.

Word

Insert Page 1 Line 5 Text Doc

L...+T...1T...+T...2T...+T...3T...+T...

This graph shows the graphical abilities of Visi On.

Multiple apps are ruining at the same time.

Word

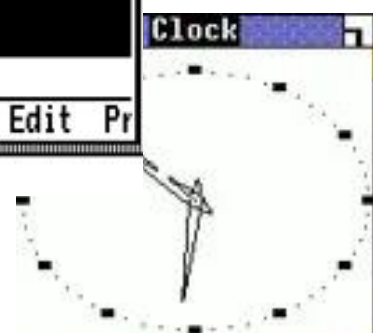
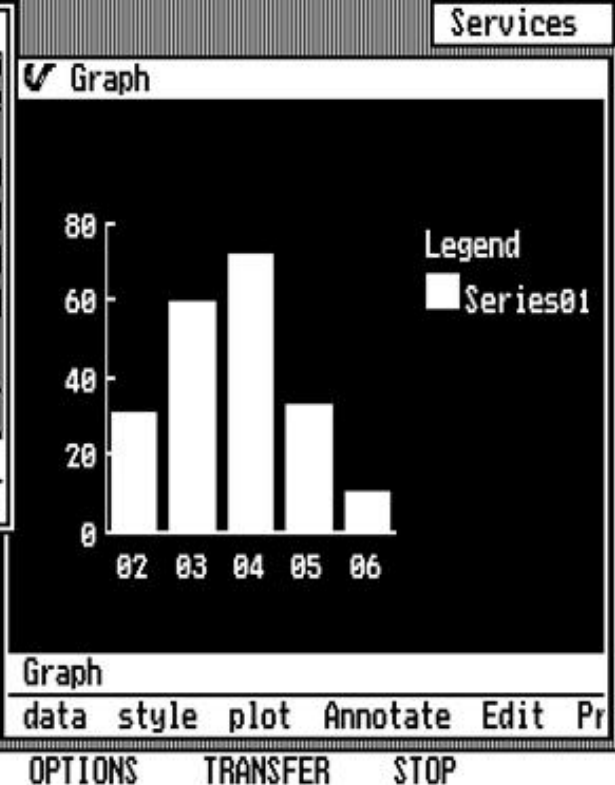
delete Cut&Paste locate enhance sty

Another Document_

Word

delete Cut&Paste locate enhance sty

HELP CLOSE OPEN FULL FRAME



MS-DOS Executive

Write - README.DOC

File View Special

A C D

C: \WINDOWS

Microsoft Windows

MS-DOS Executive

Version 1.01

Copyright © 1985, Microsoft Corp.

Ok

Disk Space Free: 30024K

Memory Free: 303K

Page 1



Hey
"Siri"



Find My
Item Sharing



Pets albums in Photos



Contact
Posters



Offline
maps



NameDrop



Swipe

to reply

Leave a
FaceTime message



Journal



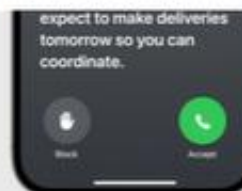
StandBy



Check In



Live Stickers



Live
Voicemail

Improved autocorrect




Chamadas de Sistemas

- Como já vimos, o SO é composto por rotinas, que compõem o kernel oferecendo serviços aos usuários e às aplicações.
- Todas as funções do kernel são implementadas por meio de rotinas do sistema que necessariamente possuem em seu código instruções de modo kernel.

Assim, as rotinas do sistema operacional não estão disponíveis para aplicações do usuário e deve ser implementado mecanismo de proteção a tais rotinas.

Cada serviço tem um *System Call* associado e cada SO tem uma biblioteca de *System Calls*.

Todo o controle de execução de rotinas do sistema operacional é realizado pelo mecanismo conhecido como *system call*.



Toda vez que uma aplicação desejar chamar uma rotina do SO, o mecanismo de *system call* é ativado.



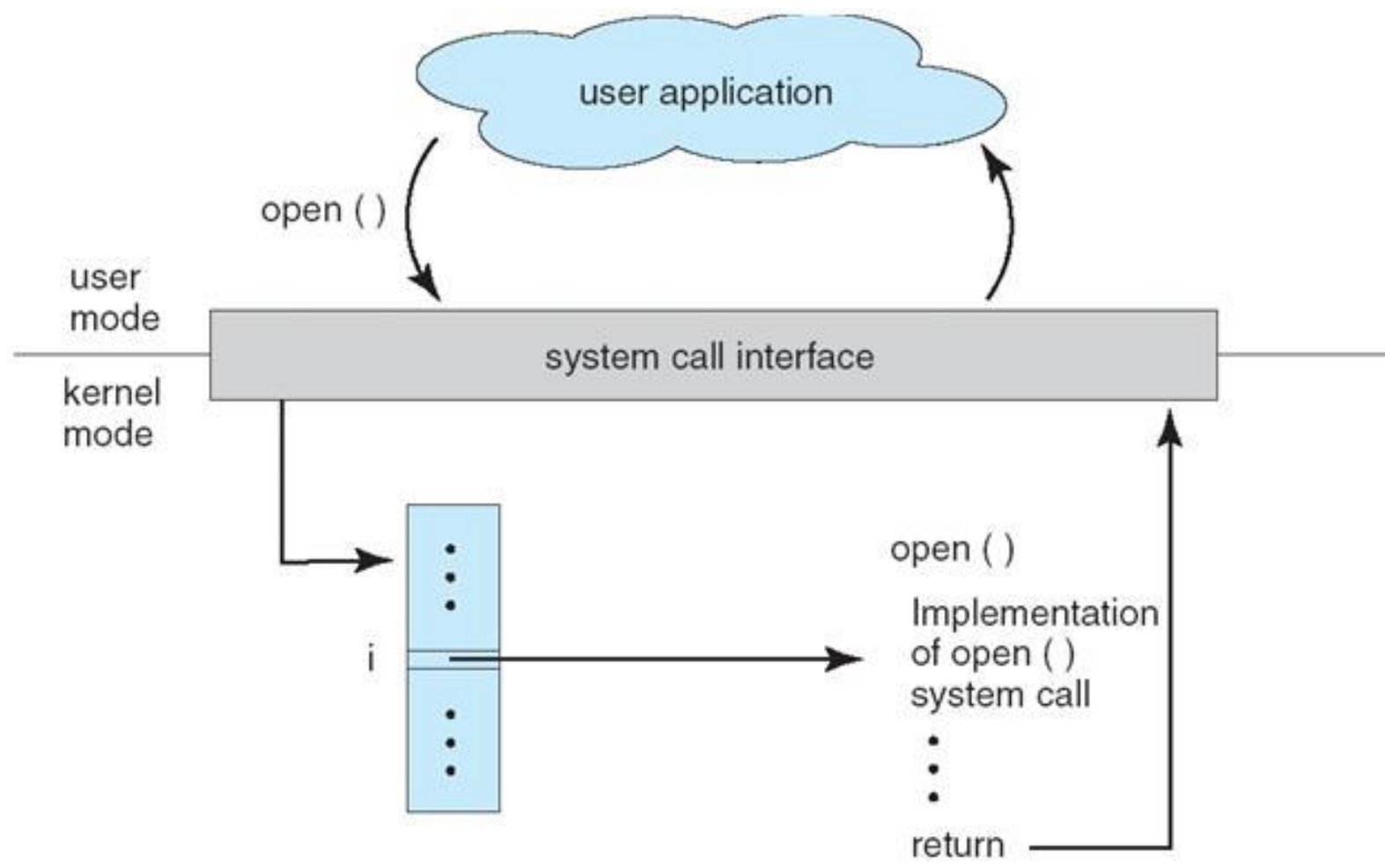
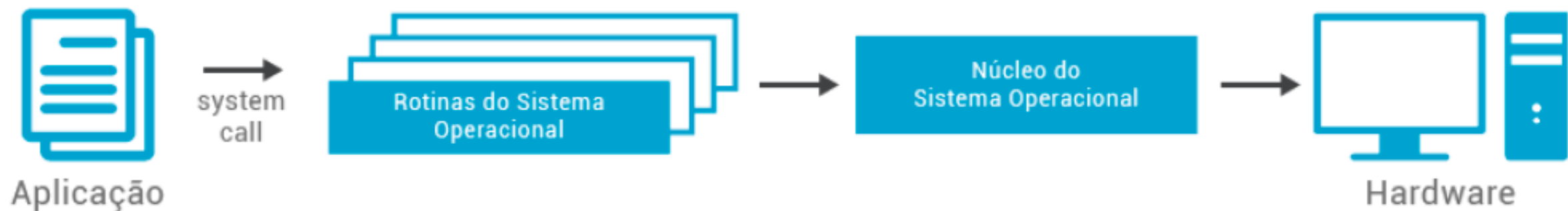
O SO verificará se a aplicação possui privilégios necessários para executar a rotina solicitada.



Em caso negativo, o SO impedirá o desvio para a rotina do sistema, sinalizando ao programa chamador que a operação não é possível.

- Se a aplicação possuir o privilégio para chamar a rotina, o SO primeiramente salva o conteúdo dos registradores, troca o modo de acesso do processador de usuário para kernel e realiza o desvio para a rotina alterando o registrador PC (Program Counter) com o endereço da rotina chamada.
- Ao terminar a execução da rotina do sistema, o modo de acesso é alterado de kernel para usuário, e também o contexto dos registradores restaurados, para que a aplicação continue a ser executada a partir da instrução que chamou a rotina do sistema.

- As rotinas do SO e as *system calls* dão acesso as instruções privilegiadas e ao núcleo do sistema.
- Sempre que uma aplicação do usuário desejar executar uma rotina do SO deve invocar a *system call* que irá executar as rotinas do sistema operacional e retornar um status para a aplicação do usuário.
- Os mecanismos de *system call* e de proteção por hardware devem garantir a segurança e a integridade do sistema.
- Dessa forma, as aplicações serão impedidas de executarem instruções privilegiadas sem autorização e a supervisão do sistema operacional



Serviços do SO

- O sistema operacional se prepara para execução de programas e requer um conjunto de serviços e funções úteis ao usuário.
- Serviços do SO:
 - **Interface do usuário:** atualmente a interface utilizada aos usuários é a GUI (Interface Gráfica com o Usuário), composta por janelas, com dispositivo de apontamentos de E/S, menus e teclado para entrada de texto;

- **Execução de programa:** o sistema precisa carregar um programa para executá-lo, e o programa precisa encerrar sua execução.
- **Operações de E/S:** o usuário não consegue controlar os dispositivos de E/S diretamente e necessita que o SO proporcione meios para realizar essas tarefas.
- **Manipulação do sistema de arquivos:** é necessário ler, gravar e renomear os arquivos e diretórios, sistemas de pesquisa por meio de nomes ou algum atributo da propriedade (extensão, data criação, aplicativo).

- **Comunicações:** em algumas situações os processos necessitam trocar informações que estão em um mesmo computador ou não.
 - Essa comunicação pode ser implementada por meio da memória compartilhada ou troca de mensagens, em que o pacote de informações é movido entre processos do SO.
- **Detecção de erros:** o SO precisa estar preparado para os possíveis erros. Eles podem ocorrer na CPU, na memória, nos dispositivos de E/S e no programa do usuário. Para cada tipo de erro, o SO deve ter uma ação adequada para resolvê-lo.

- Existem também as funções do SO que não são usadas para auxiliar o usuário, mas para garantir sua eficiência:
 - **Alocação de recursos:** com a multiprogramação, alguns recursos podem ser solicitados simultaneamente. O SO precisa alocar recursos a cada um deles. É necessário conhecer a tarefa para que se possa adequar o melhor uso de CPU, dispositivos de E/S e memória.
 - **Contabilidade:** os contadores normalmente são destinados para pesquisadores, com o intuito de descobrir quais são os usuários e quanto utilizam dos recursos, com o objetivo de configuração e melhoria do sistema.

- **Proteção e segurança:** quando existem diversos usuários fazendo uso do mesmo sistema, há a necessidade da proteção e a não interferência entre processos.
 - A proteção permite a garantia de controle de todo acesso aos recursos do sistema, iniciando-se com a identificação dos usuários, normalmente por meio de senhas.

Arquiteturas do Kernel

- O projeto de um sistema operacional é realizado de forma a atender a requisitos operacionais de desempenho, portabilidade, confiabilidade, facilidade de uso, segurança.
- O projeto do sistema operacional depende da arquitetura de hardware e do tipo de SO que se deseja construir: batch, tempo compartilhado, tempo real, multiprogramado, multiprocessado, etc.

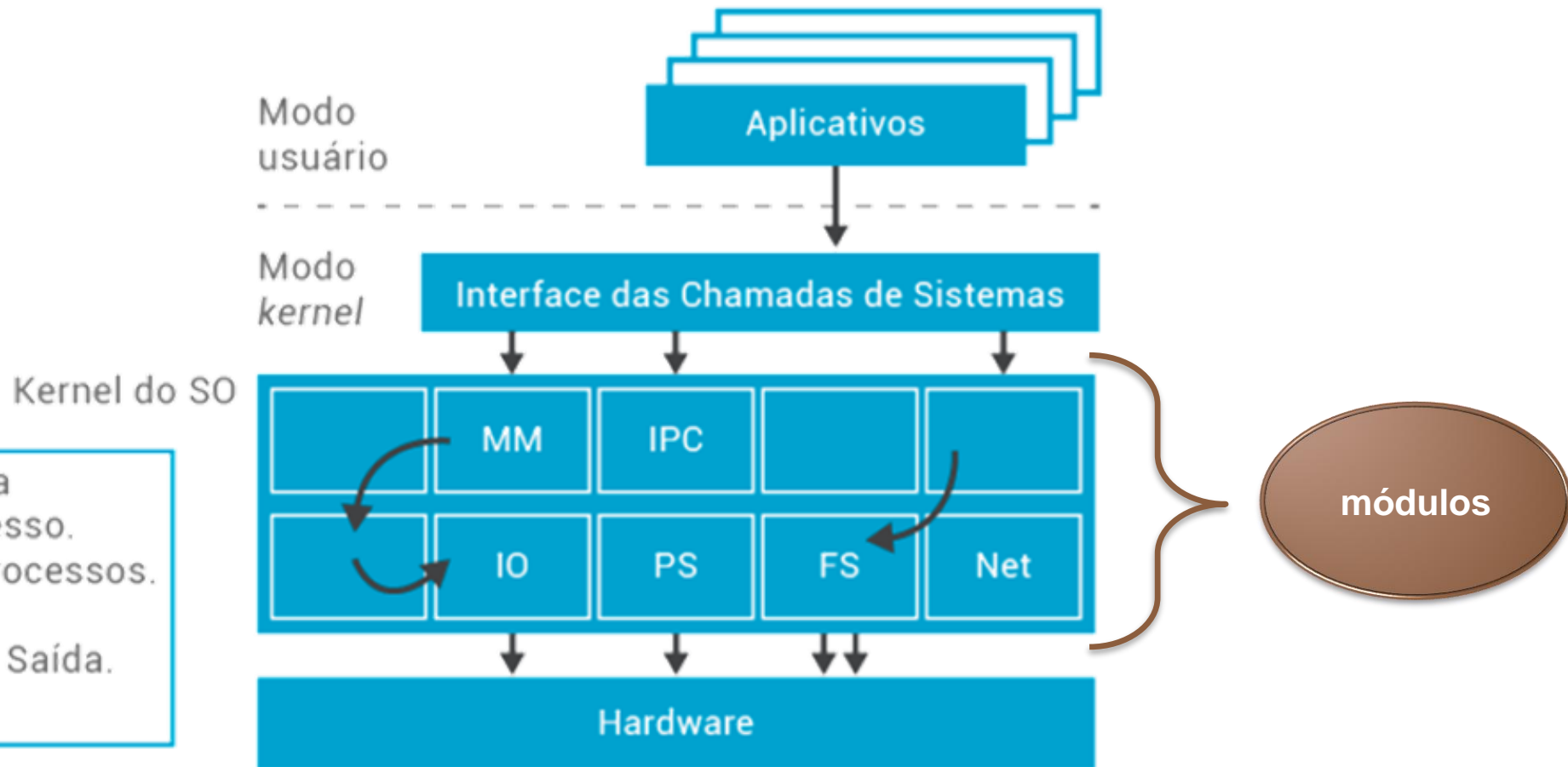
- Com a evolução dos computadores, o gerenciamento que o SO desenvolvia se tornou mais complexo pois haviam as possibilidades de multiprocessamentos com *time sharing*.
- Para que esse suporte fosse feito da maneira mais eficiente possível, foram desenvolvidas no núcleo 3 tipos de arquiteturas:
 - **Sistema Monolítico;**
 - **Sistema em Camadas;**
 - **Sistema Microkernel.**

Sistema Monolítico

- A arquitetura monolítica foi utilizada nos primeiros sistemas operacionais, como o MS-DOS e nas primeiras versões do Linux.
- Cada componente é contido no Kernel.
- Rotinas podem interagir livremente entre si.
- *System Calls* disparam o processo.

- Nesta arquitetura os componentes do sistema são compilados em módulos separados e depois “linkados”, formando um único programa executável em que os módulos podem interagir entre si livremente.
- Os módulos são carregados em memória e interagem entre si. A manutenção deste tipo de sistema é bem difícil.

MM - Gerência de Memória
PS - Escalonador de Processo.
IPC - Comunicação Interprocessos.
FS - Sistema de Arquivos
IO - Gerência de Entrada e Saída.
Net - Gerência de Redes.

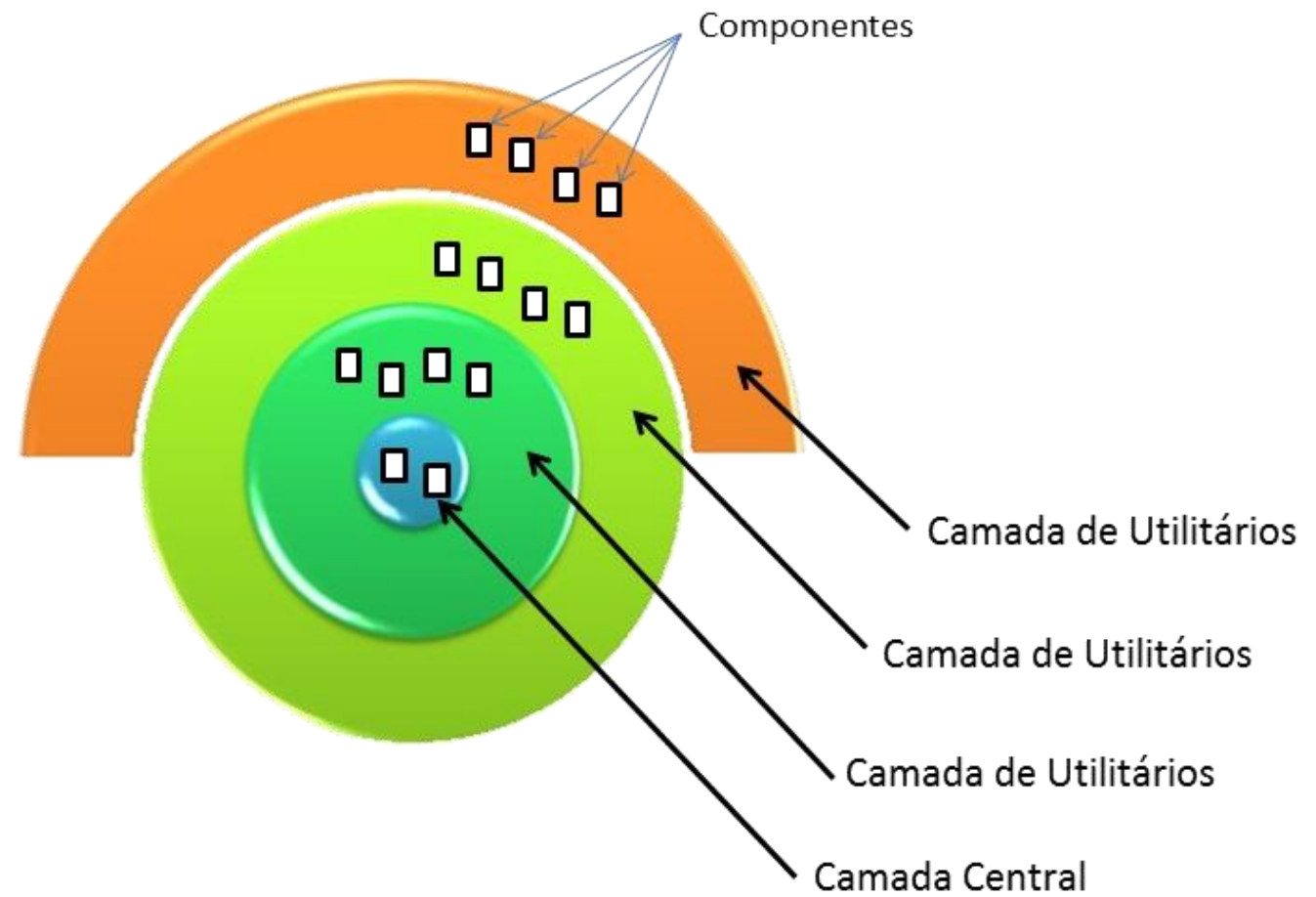
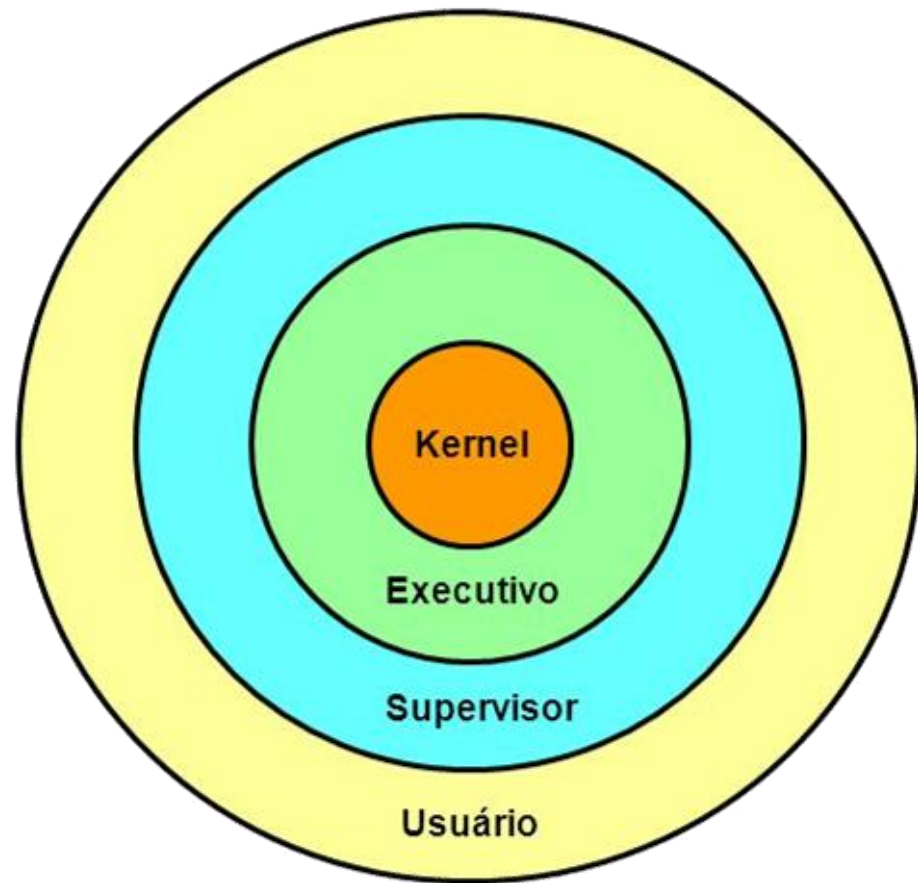


Sistemas em Camadas

- Nessa arquitetura o sistema é dividido em camadas sobrepostas.
- A arquitetura em camadas surgiu devido a complexidade dos sistemas operacionais na medida em que foram evoluindo.
- Cada camada recebe um conjunto de funções que podem ser utilizadas apenas em camadas superiores ou inferiores. As camadas inferiores são privilegiadas.

- **Vantagem:** isola as funções do SO, facilitando sua manutenção e depuração, e também cria uma hierarquia de níveis de modo de acesso, protegendo as camadas mais internas.
- **Desvantagem:** desempenho fica comprometido, pois a camada só pode se comunicar com a camada acima ou com a camada abaixo.
 - Quando um aplicativo do usuário solicita um serviço da camada kernel é necessário passar por várias outras camadas e realizar várias trocas do modo de acesso.



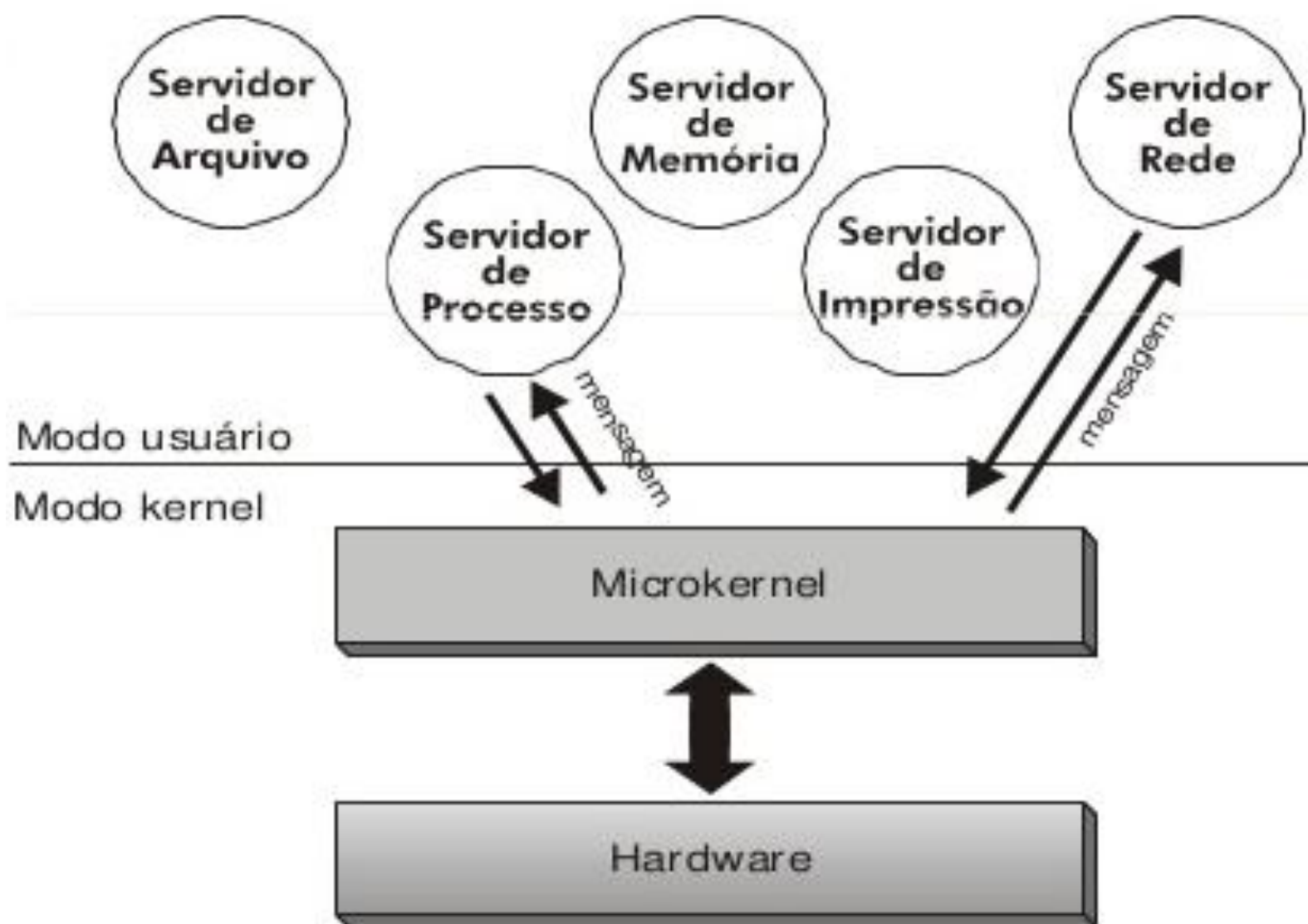


Microkernel (Cliente Servidor / Client Server)

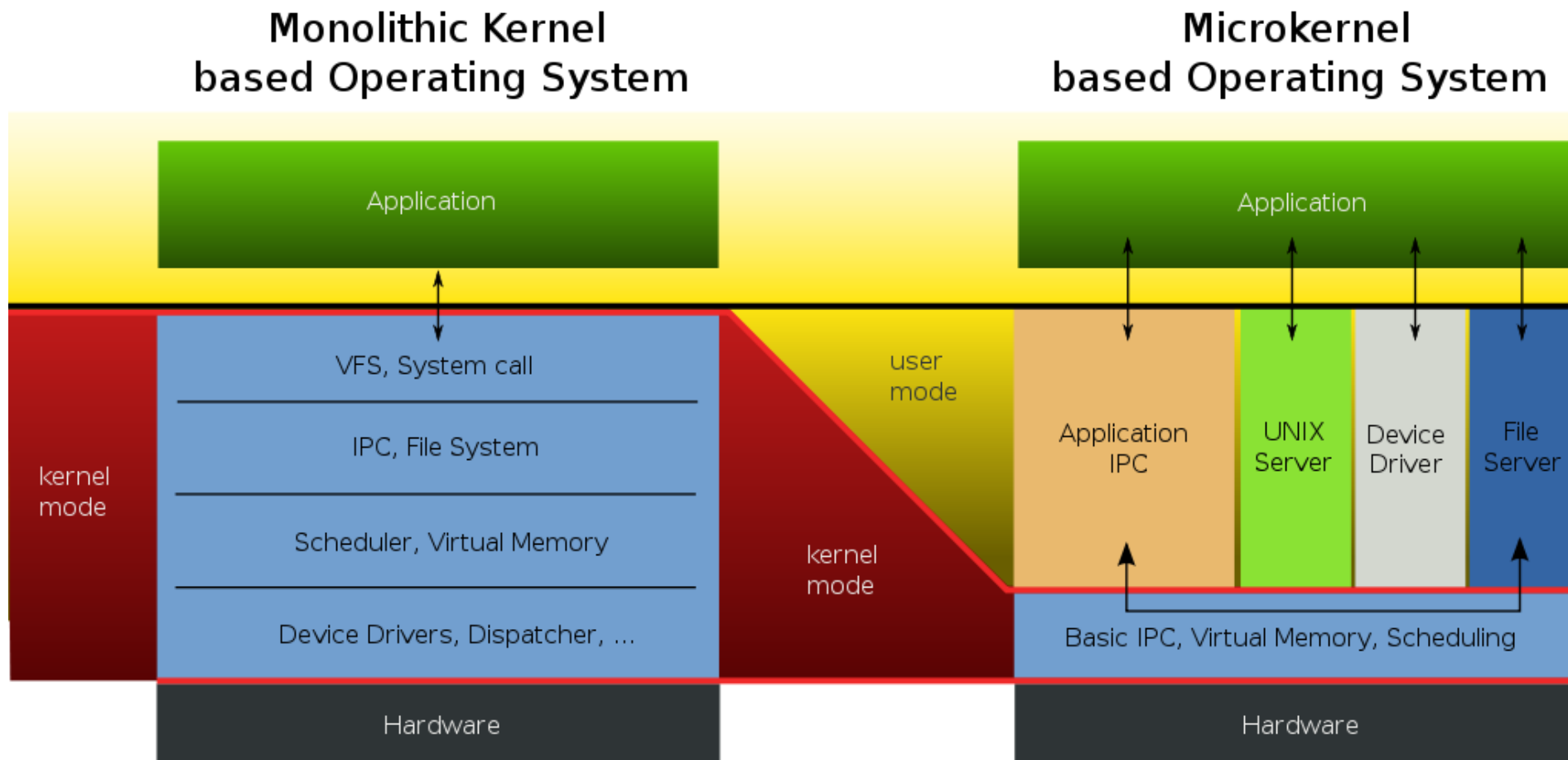
- O núcleo é menor e o mais simples possível, os sistemas são oferecidos por meio de processos, em que cada um é responsável por oferecer um conjunto específico de funções.
- Assim, se um serviço parar o sistema, não fica comprometido, por exemplo: em um sistema microkernel existem os servidores – arquivos, impressão e internet. Se, por algum motivo, o servidor de internet parar, o serviço cessa, mas a rede não.

- Quando uma aplicação do usuário solicita um serviço é feita uma solicitação ao processo responsável pelo serviço.
- A aplicação que solicita o serviço é chamada cliente e o processo que responde à solicitação é chamado de servidor.
- Para que essa comunicação seja realizada, é utilizado o sistema de troca de mensagem.
- A principal função do núcleo é realizar a comunicação, ou seja, a troca de mensagens entre cliente e servidor.

- Nesta arquitetura, os processos executam suas funções em modo usuário, ou seja, não tem acesso a instruções privilegiadas, não tem acesso aos componentes do sistema.
- Apenas o núcleo executa em modo kernel. Isto garante que caso haja um erro em um processo o sistema não ficará completamente comprometido, aumentando a disponibilidade do sistema.



Inter-Process Communication (IPC): comunicação entre processos



Arquitetura do Kernel dos Principais SOs

- **Windows 10/11**
- **Arquitetura: Híbrida**
- O kernel do Windows 10/11 usa uma arquitetura híbrida, que combina características dos kernels monolíticos e microkernels.
- Ele integra funcionalidades essenciais no kernel, como o gerenciamento de processos e memória, mas também separa algumas funcionalidades em subsistemas, como drivers e outros serviços o que permite um bom equilíbrio entre desempenho e modularidade.

- **Linux**
- **Arquitetura: Monolítica**
- O kernel do Linux é monolítico, ele inclui o máximo possível de funcionalidades dentro do próprio kernel, como o gerenciamento de processos, memória, sistemas de arquivos, drivers de hardware, etc.
- No entanto, ele é modular, permitindo a adição ou remoção de módulos em tempo de execução, o que oferece flexibilidade.

- **Unix**
- **Arquitetura: Principalmente Monolítica (com variantes)**
- O Unix tradicionalmente usa uma arquitetura monolítica, onde o kernel inclui todas as funcionalidades básicas de gerenciamento do sistema.
- No entanto, versões mais modernas de Unix, como o Solaris, implementaram características de microkernels, mas ainda mantêm uma estrutura monolítica básica. Cada variante de Unix pode ter diferenças sutis na implementação de seu kernel.

- **macOS**
- **Arquitetura: Híbrida**
- O kernel do macOS, conhecido como XNU (X não é de Unix), tem uma arquitetura híbrida.
- O XNU é uma combinação de três componentes principais:
 - Mach: Um microkernel que lida com multitarefa, comunicação entre processos, e abstração de hardware.
 - BSD: Uma camada de Unix que oferece funcionalidades tradicionais de sistemas operacionais, como o gerenciamento de processos e sistemas de arquivos.
 - I/O Kit: Um framework orientado a objetos que facilita a implementação de drivers.
- Essa arquitetura híbrida permite que o macOS combine a modularidade e robustez do microkernel com a eficiência e familiaridade da camada Unix.

- **Android**
- **Arquitetura: Monolítica (baseada no kernel do Linux)**
- O Android utiliza um kernel monolítico baseado no kernel do Linux. Isso significa que o kernel do Android inclui várias funcionalidades essenciais dentro do próprio núcleo, como gerenciamento de processos, memória, sistemas de arquivos e drivers de hardware.
- No entanto, o Android adapta o kernel do Linux com algumas modificações para atender às necessidades específicas de dispositivos móveis, como a inclusão do Android-specific power management e o Binder IPC (Inter-Process Communication). Ele também suporta módulos de kernel para permitir alguma modularidade.

- **iOS**
- **Arquitetura: Híbrida (baseada no kernel XNU)**
- O iOS usa o kernel XNU, que também é utilizado no macOS. Assim como no macOS, o XNU é uma arquitetura híbrida.
- No iOS, o XNU é otimizado para dispositivos móveis, com melhorias voltadas para eficiência energética, segurança e suporte a hardware específico, como a arquitetura ARM.

Máquina Virtual (Virtual Machine)



Um sistema operacional é formado por níveis, onde a camada de nível mais baixo é o hardware.



Acima dessa camada temos o sistema operacional que oferece serviços para os aplicativos do usuário.

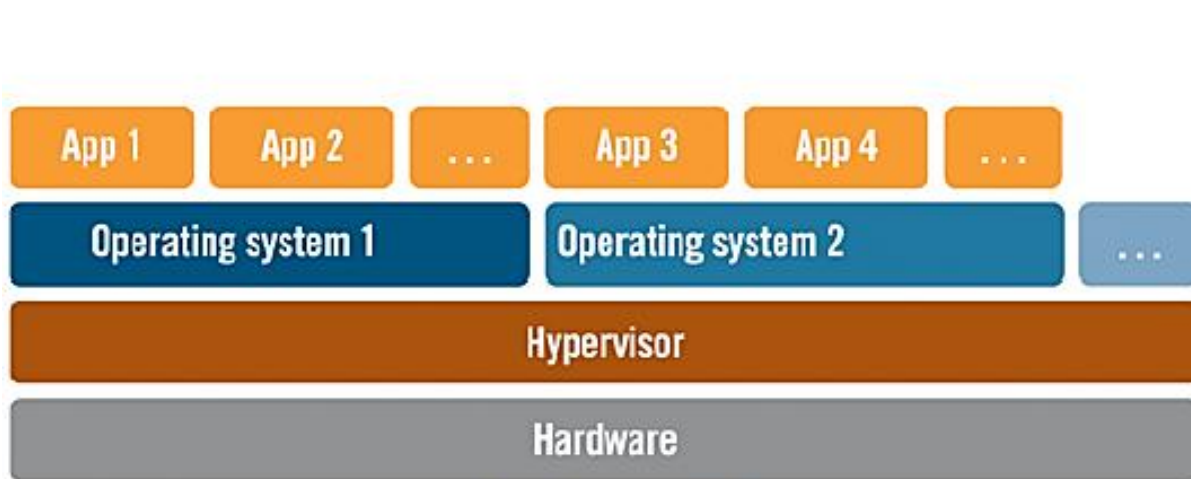
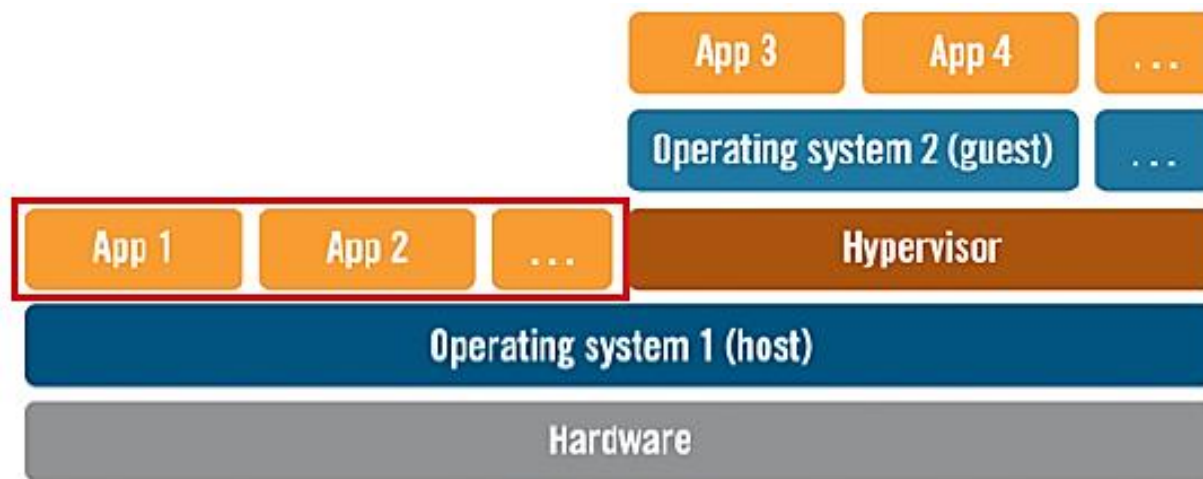


Na arquitetura de máquina virtual existe uma camada intermediária entre o hardware e o sistema operacional chamada gerência de máquinas virtuais (hypervisor).

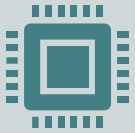


Esta camada cria diversas máquinas virtuais independentes, onde cada uma oferece uma cópia virtual do hardware, incluindo os modos de acesso, interrupções, memória, dispositivos de entrada e saída, etc.

- Como cada máquina virtual é independente das outras, é possível que cada VM tenha seu próprio sistema operacional e que seus usuários executem suas aplicações como se o computador estivesse dedicado a cada um deles.
- Cada máquina virtual é isolada das demais o que proporciona segurança para cada VM. Isto garante também confiabilidade pois uma VM não pode comprometer o estado das outras VMs.

**Tipo Bare-Metal****Tipo Hosted**

Virtualização Bare-Metal

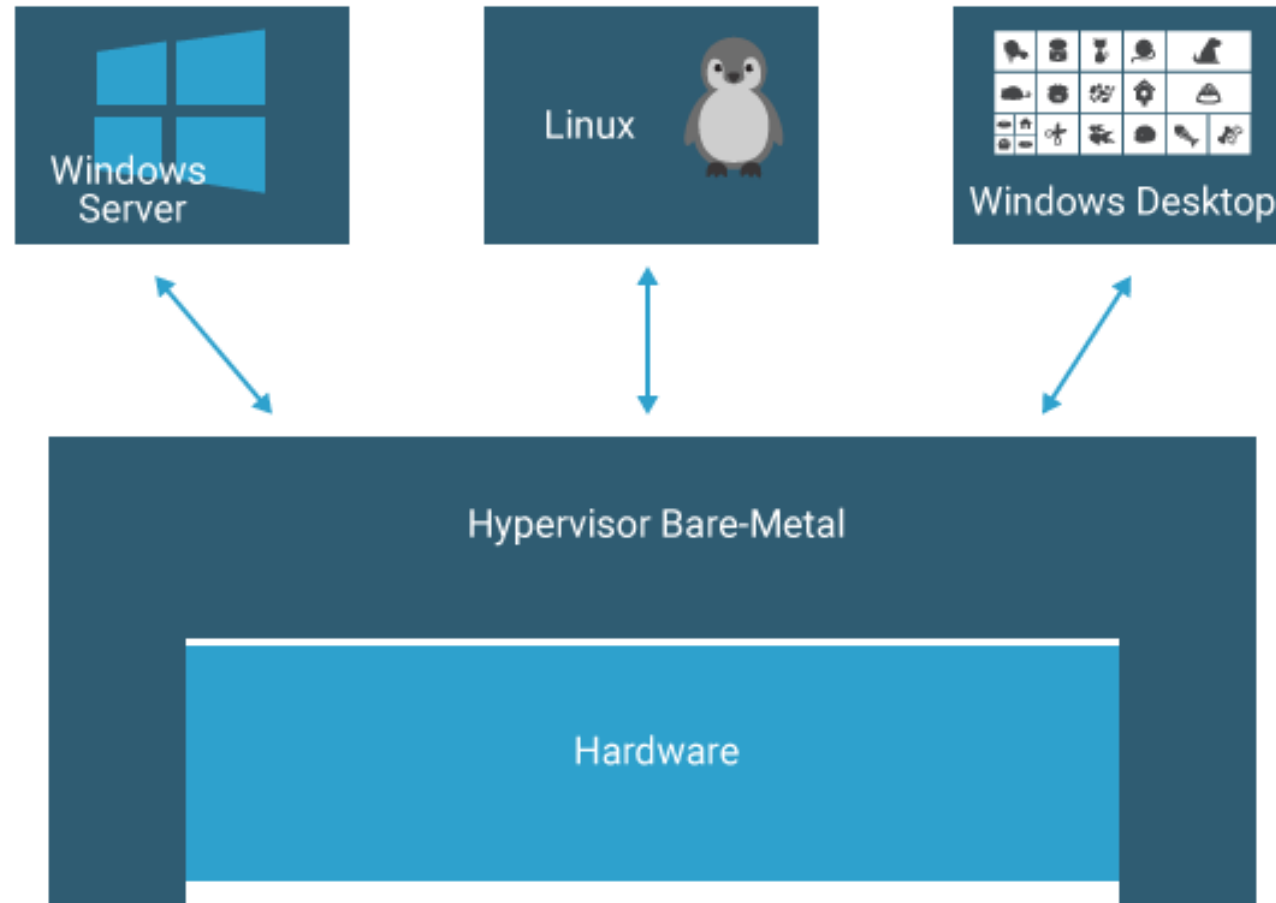


O hypervisor está instalado diretamente no hardware físico e, depois, outros sistemas operacionais e aplicativos são instalados.



Isso fez com que os hypervisors fossem chamados de sistemas operacionais, pelo sentido funcional da coisa.

BARE-METAL



Host Virtualization ou Hosted

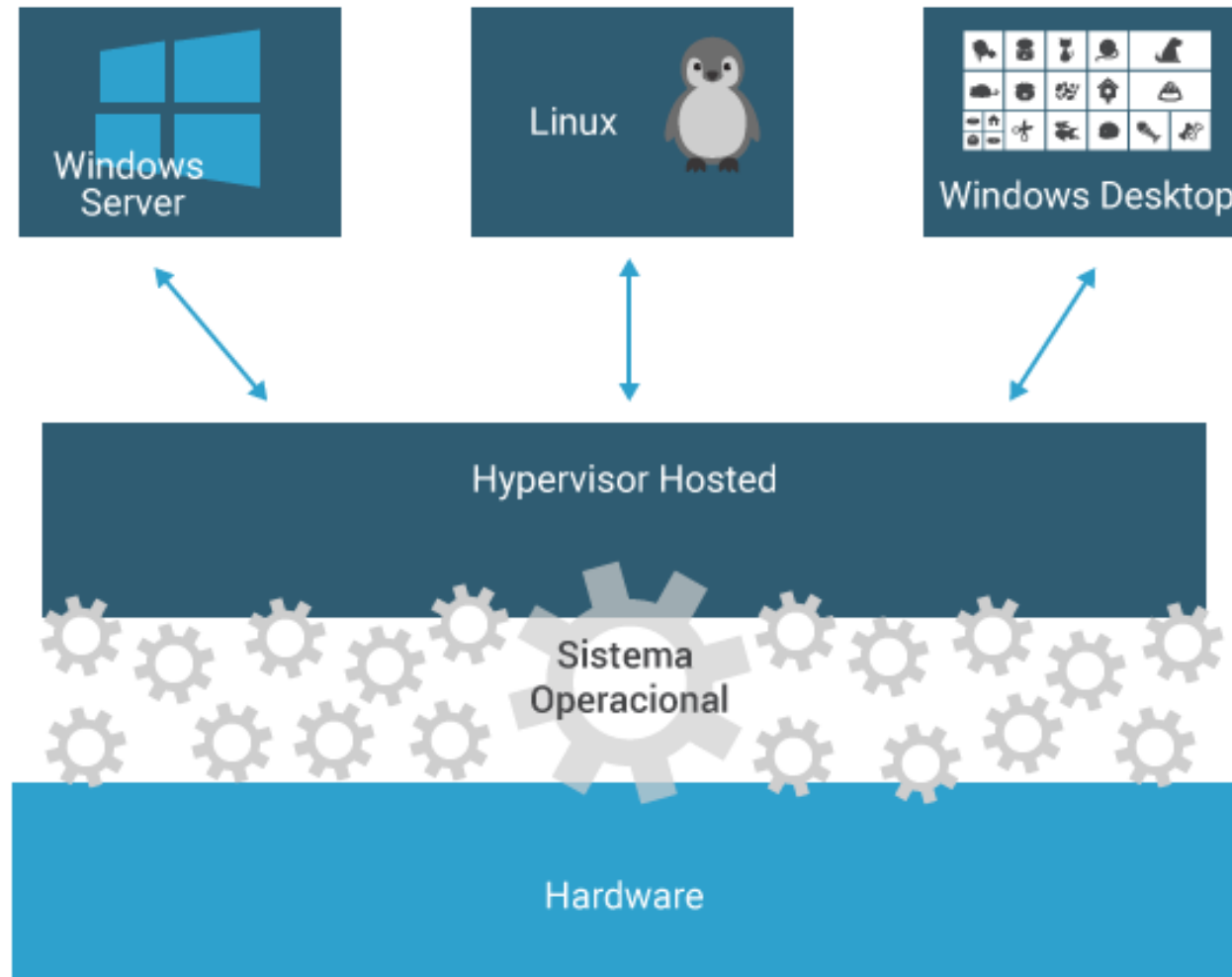


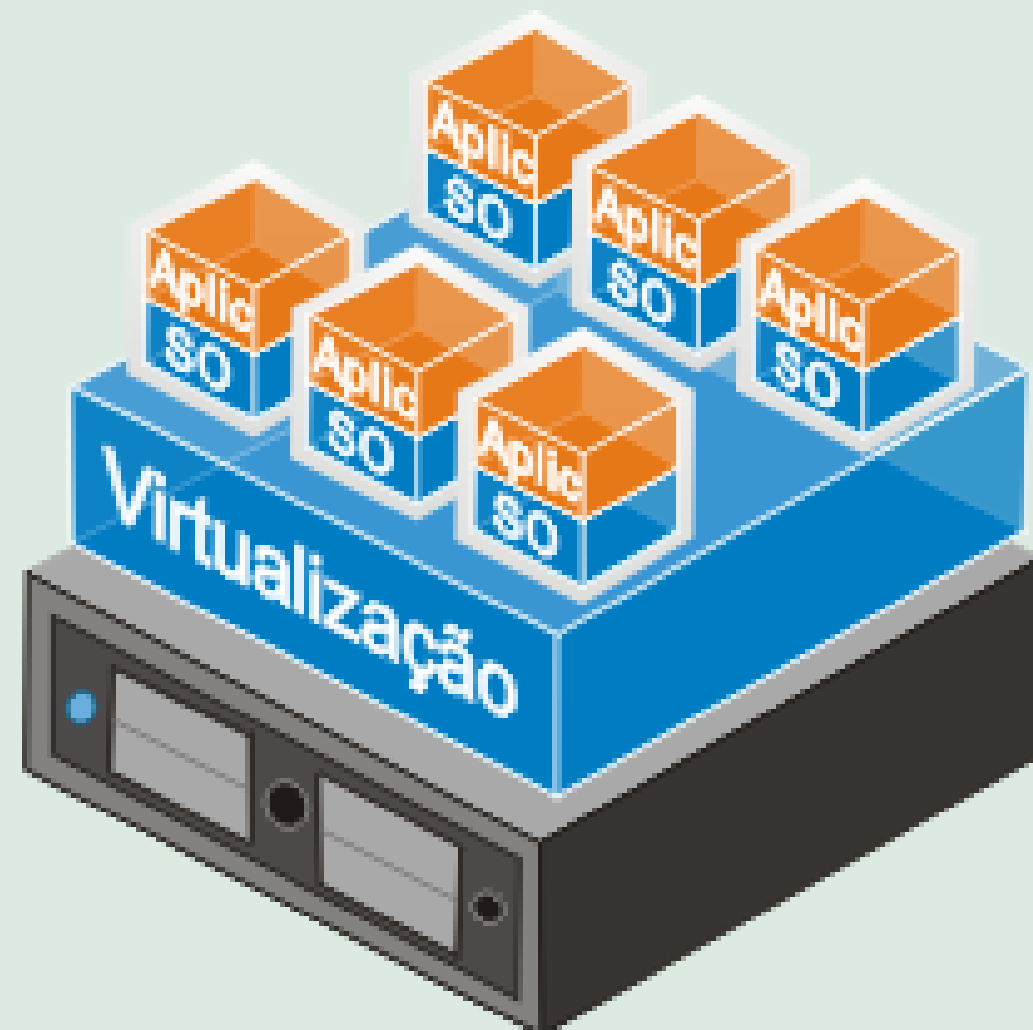
Envolve instalar um sistema operacional no hardware e instalar o hypervisor neste sistema operacional.



Essa solução não é tão usada quando a bare-metal.

HOST VIRTUALIZATION



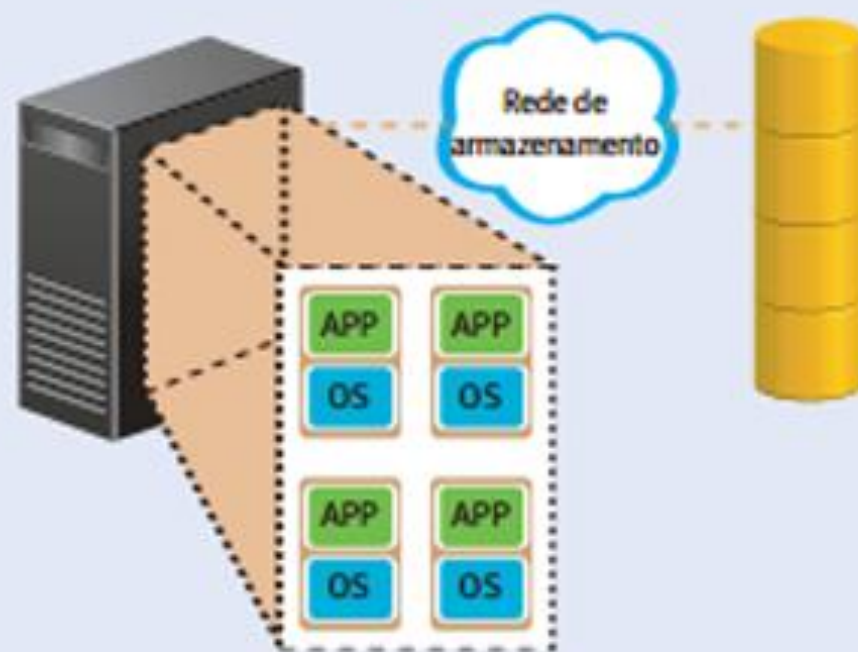


Antes

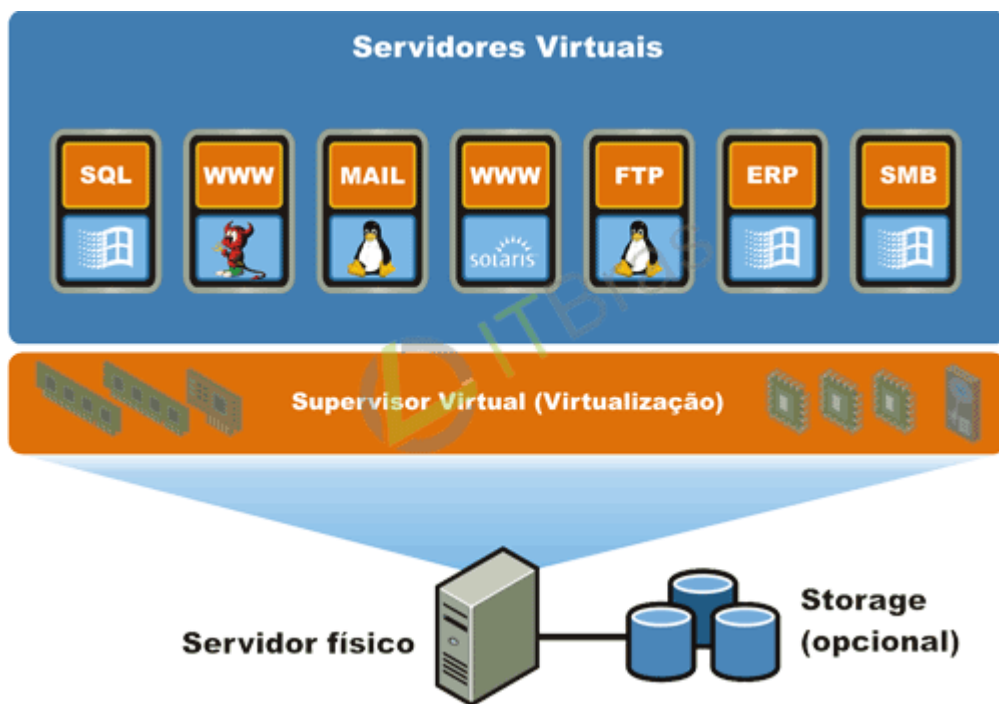


Os aplicativos são dedicados a um hardware de servidor específico.

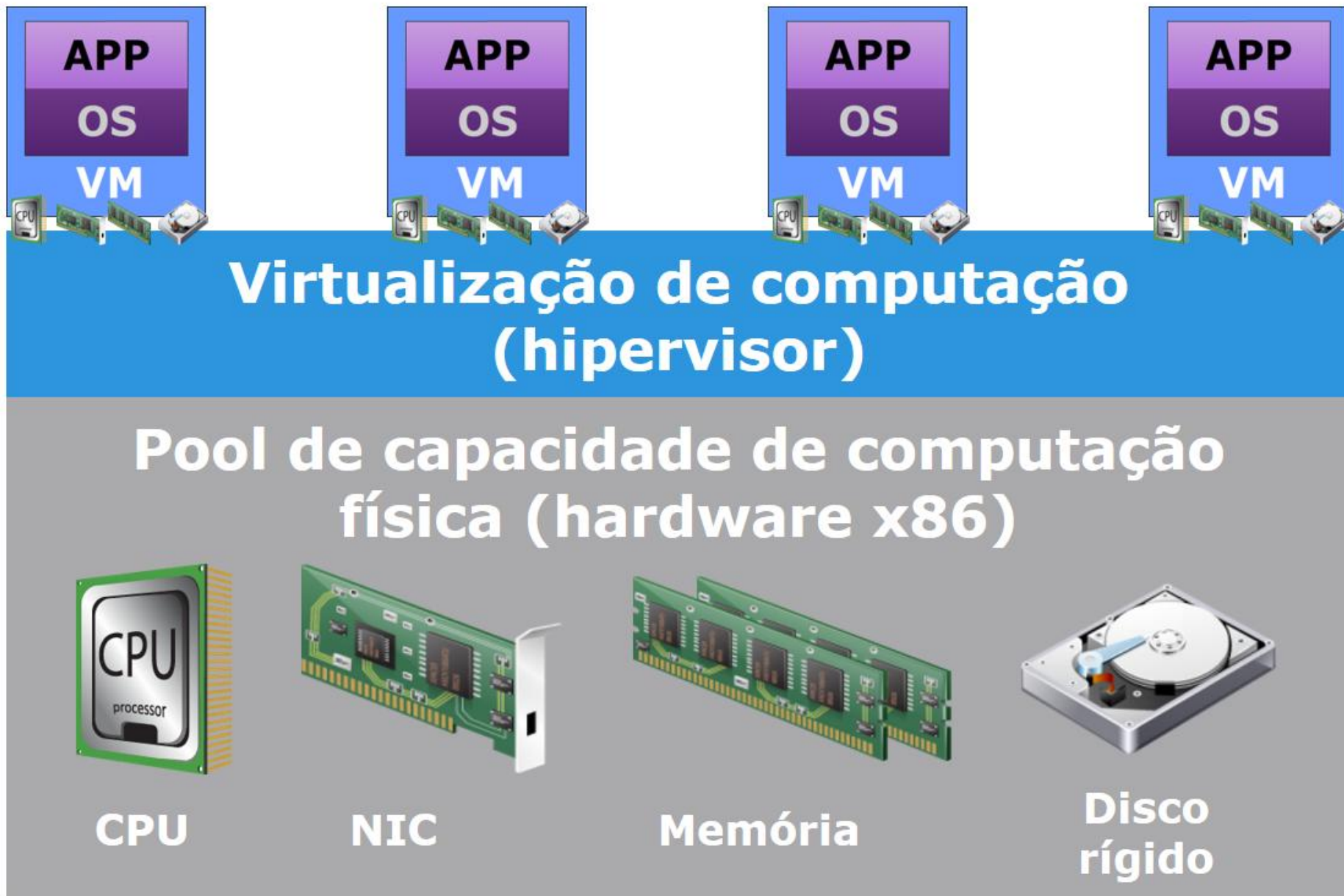
Depois



Os aplicativos residem em computadores virtuais que compartilham os recursos de armazenamento e de computadores



Desacoplado





VÍDEOS

PRODUCTION

DIRECTOR

CAMERA

SCENE

TAKE

- **História dos Sistemas operacionais**

<https://www.youtube.com/watch?v=9rC9GilX1Io> 4:54

- **Interface Gráfica**

<https://www.youtube.com/watch?v=Vd0A2fjxR4A> 6:24

- **Fundamentos da virtualização**

<https://www.youtube.com/watch?v=-9fcJ8KVeuw> 3:13