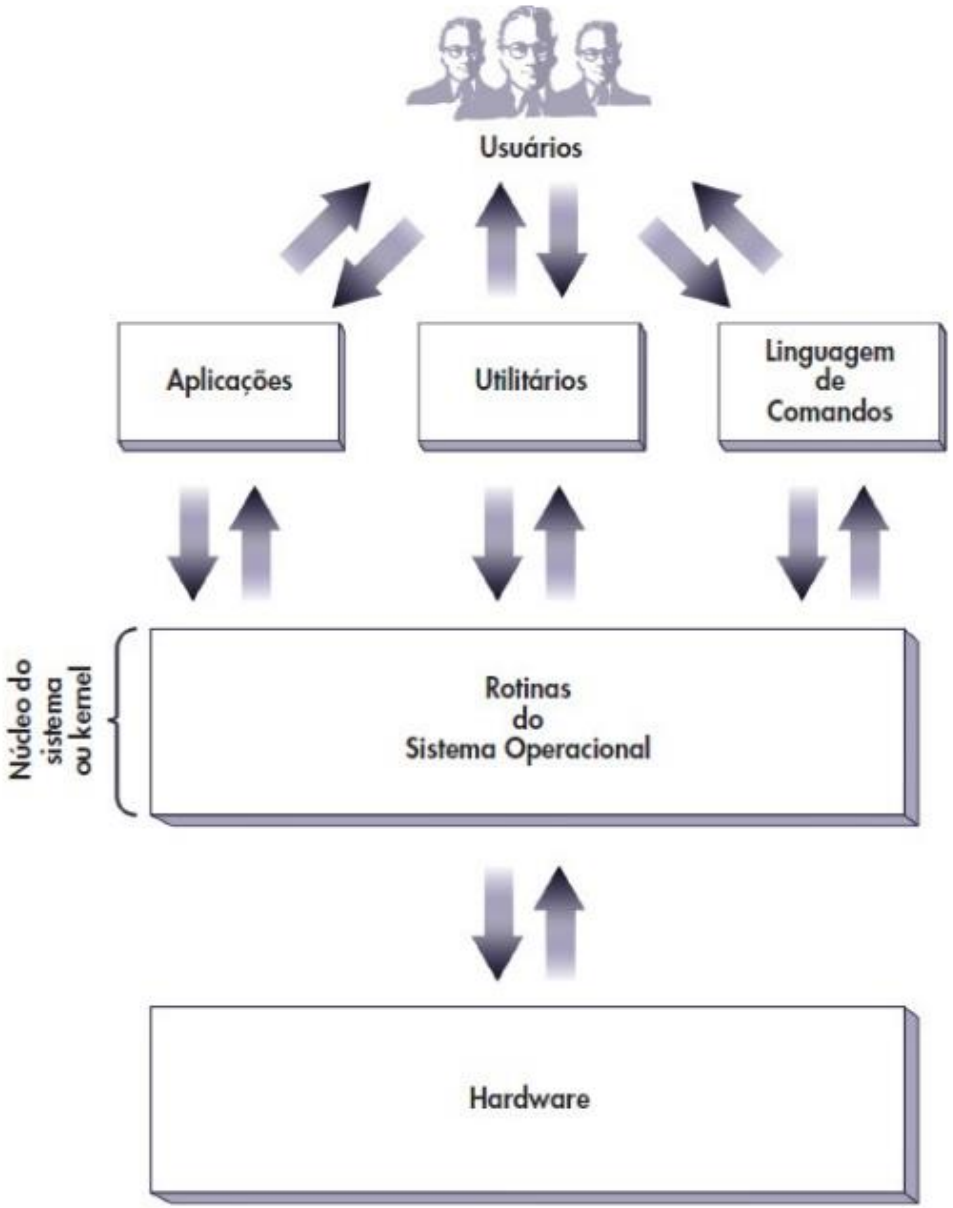
Estrutura do Sistema Operacional

Professor Danilo

Funções do Núcleo

- O sistema operacional é formado por um **conjunto de rotinas que oferece serviços aos usuários e às suas aplicações**.
- Esse conjunto de rotinas é denominado **núcleo do sistema, ou kernel**.
- A maioria dos sistemas operacionais é fornecida acompanhada de utilitários e linguagem de comandos, que são ferramentas de apoio ao usuário, porém não são parte do núcleo do sistema.

Estrutura do sistema operacional



Funções do Núcleo

- Diferentemente de uma aplicação convencional, com sequenciamento de início, meio e fim, as rotinas do sistema são executadas concorrentemente sem uma ordem predefinida (**assíncrona**).

Principais funções do núcleo

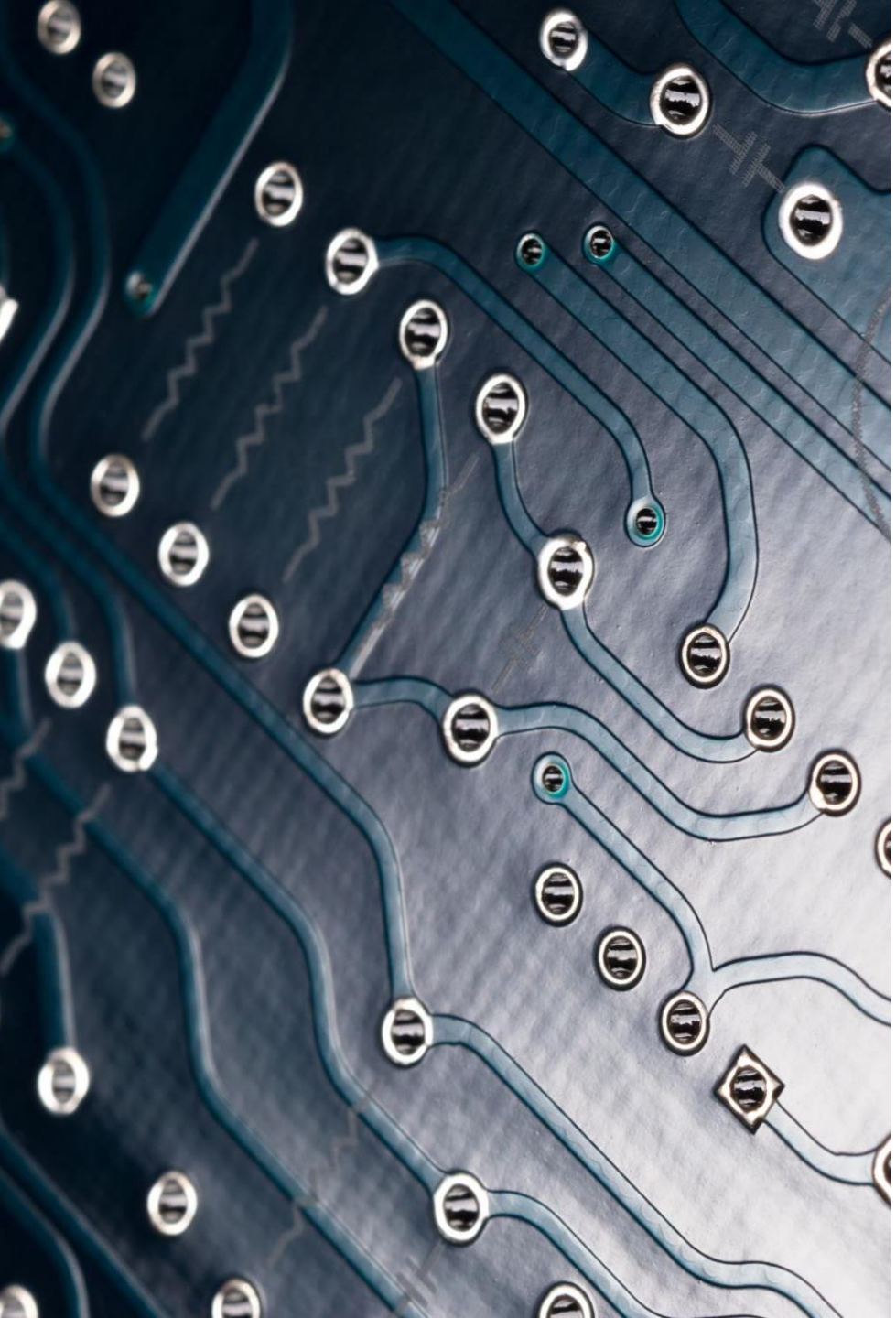
- Tratamento de interrupções e exceções;
- Criação e eliminação de processos e threads;
- Sincronização e comunicação entre processos e threads;
- Escalonamento e controle dos processos e threads;
- Gerência de memória;
- Gerência do sistema de arquivos;
- Gerência de dispositivos de E/S;
- Suporte a redes locais e distribuídas;
- Contabilização do uso do sistema;
- Auditoria e segurança do sistema.

Modo de Acesso

- **Para solucionar diversos problemas originados pelo ambiente multiprogramável**, o sistema operacional deve implementar **mecanismos de proteção que controlem o acesso concorrente aos diversos recursos do sistema.**

Modo de Acesso

- Uma preocupação que surge nos projetos de sistemas operacionais é a implementação de **mecanismos de proteção ao núcleo do sistema e de acesso aos seus serviços**.
- Caso uma aplicação, que tenha acesso ao núcleo, realize uma operação que altere sua integridade, todo o sistema poderá ficar comprometido e inoperante.
- Muitas das principais **implementações de segurança de um sistema operacional** utilizam um **mecanismo presente no hardware dos processadores**, conhecido como **modo de acesso**.



Modo de Acesso

- Em geral, os processadores possuem dois **modos de acesso: modo usuário e modo kernel**.
- Quando o processador trabalha no modo usuário, uma aplicação só pode executar instruções conhecidas como não privilegiadas, tendo acesso a um número reduzido de instruções, enquanto no modo kernel a aplicação pode ter acesso ao conjunto total de instruções do processador.
- O **modo de acesso** é determinado por um conjunto de bits, **localizado no registrador de status do processador**, que indica o modo de acesso corrente. Por intermédio desse registrador, **o hardware verifica se a instrução pode ou não ser executada**.

Modo de Acesso

- As instruções privilegiadas não devem ser utilizadas de maneira indiscriminada pelas aplicações, pois isso poderia ocasionar sérios problemas à integridade do sistema.
- **Suponha que uma aplicação atualize um arquivo em disco. O programa, por si só, não deve especificar diretamente as instruções que acessam seus dados no disco. Como o disco é um recurso compartilhado, sua utilização deverá ser gerenciada unicamente pelo sistema operacional, evitando que a aplicação possa ter acesso a qualquer área do disco sem autorização, o que poderia comprometer a segurança e a integridade do sistema de arquivos.**

Modo de Acesso

- Certas instruções só devem ser executadas pelo sistema operacional ou sob sua supervisão, impedindo problemas de segurança e integridade do sistema.
- **As instruções privilegiadas só podem ser executadas quando o modo de acesso do processador encontra-se em kernel**, caso contrário o hardware irá impedir a execução da instrução.
- As instruções não privilegiadas são as que não oferecem risco ao sistema e podem ser executadas em modo não privilegiado, ou seja, modo usuário.

Rotinas do Sistema Operacional e System Calls

- **As rotinas do sistema operacional compõem o núcleo do sistema, oferecendo serviços aos usuários e suas aplicações**.
- Todas as funções do núcleo são implementadas por rotinas do sistema que necessariamente **possuem em seu código instruções privilegiadas**. A partir desta condição, para que estas rotinas possam ser executadas o processador deve estar obrigatoriamente em modo kernel, o que exige a implementação de mecanismos de proteção para garantir a confiabilidade do sistema.

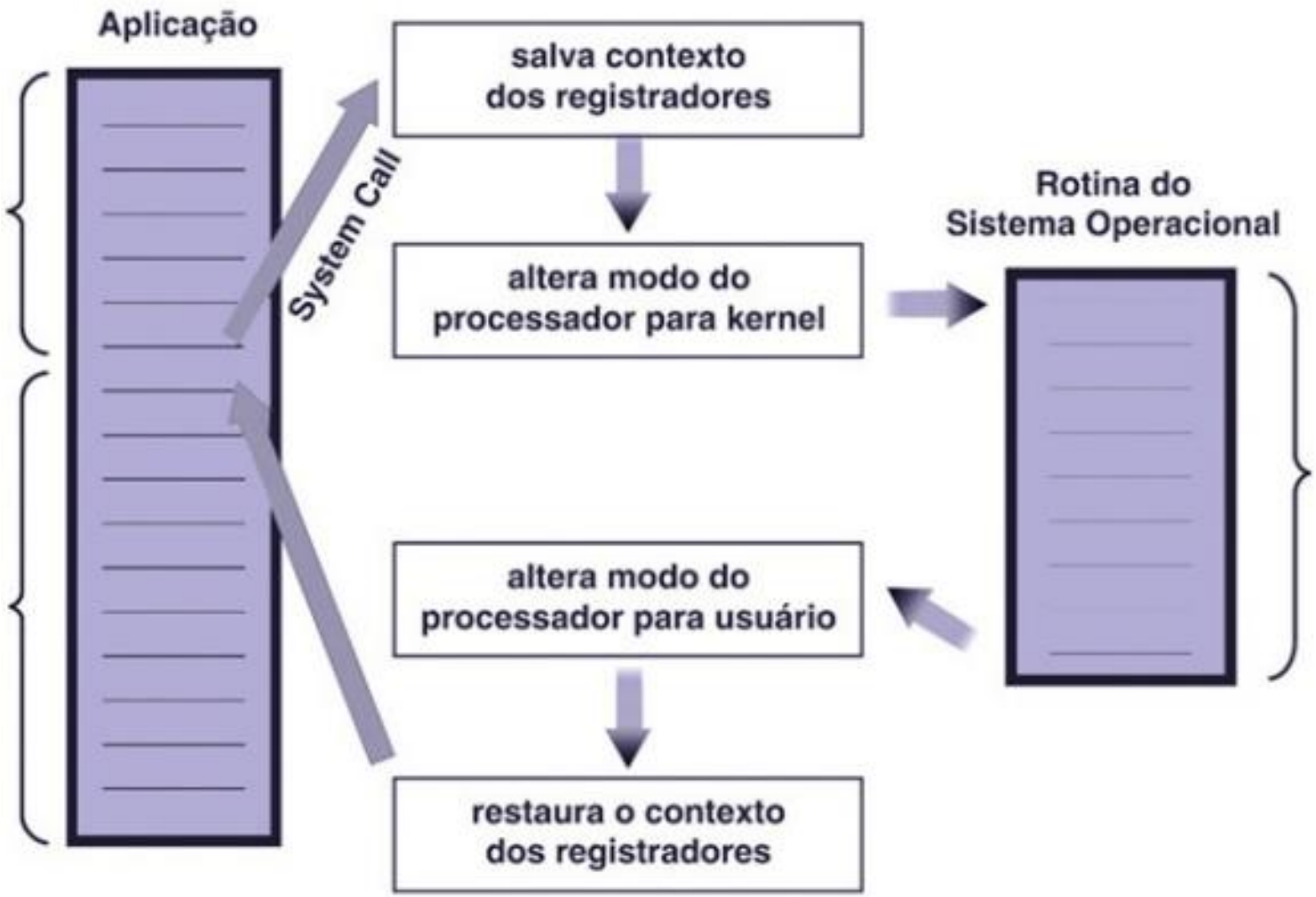
System Call

- **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 sistema operacional, o mecanismo de system call é ativado.
- Inicialmente**, o sistema** operacional **verificará se a aplicação possui privilégios necessários para executar a rotina desejada**. Em caso negativo, o sistema operacional impedirá o desvio para a rotina do sistema, sinalizando ao programa chamador que a operação não é possível.

System Call

- Este é um mecanismo de proteção por software, no qual o sistema operacional garante que as aplicações só poderão executar rotinas do sistema que estão previamente autorizadas.

Chamada a uma rotina do sistema



System Call

- **Uma aplicação sempre deve executar com o processador em modo usuário**.
- Caso uma aplicação tente executar diretamente uma instrução privilegiada sem ser por intermédio de uma chamada à rotina do sistema, um mecanismo de proteção por hardware garantirá a segurança do sistema, impedindo a operação (será sinalizado um erro).

System Call

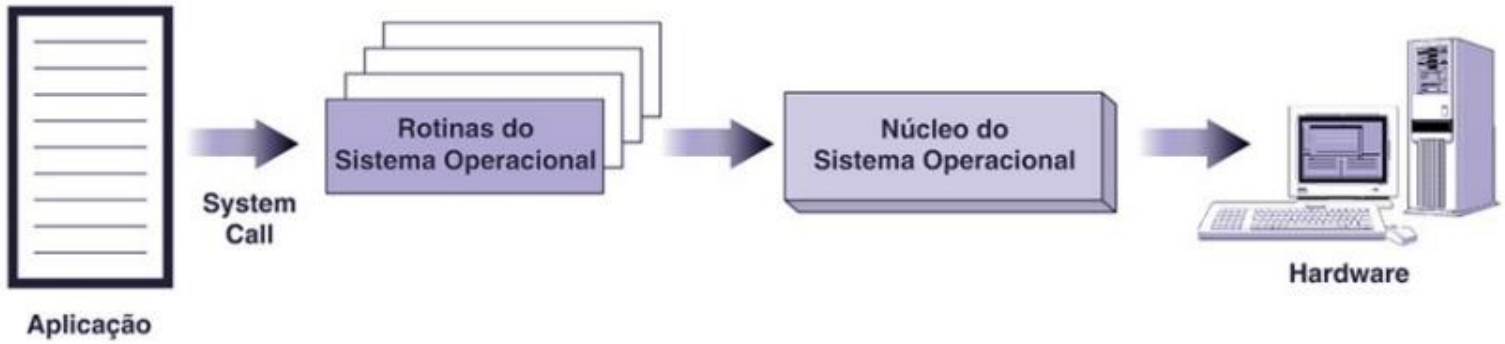
- Os mecanismos de system call e de proteção por hardware **garantem a segurança e a integridade do sistema**. Com isso, as aplicações estão impedidas de executarem instruções privilegiadas sem a autorização e a supervisão do sistema operacional.

Chamada a Rotinas do Sistema Operacional

As **rotinas do sistema e** o mecanismo de **system call** podem ser entendidos como uma **porta de entrada para o núcleo do sistema** operacional e a seus serviços.

Sempre que uma aplicação desejar algum serviço do sistema, deve ser realizada uma chamada a uma de suas rotinas através de uma system call.

Chamada a uma rotina do sistema



Funções das system calls

Gerência de processos e threads	Criação e eliminação de processos e threads
	Alteração das características de processos e threads
	Sincronização e comunicação entre processos e threads
	Obtenção de informações sobre processos e threads
Gerência de memória	Alocação e desalocação de memória
Gerência do sistema de arquivos	Criação e eliminação de arquivos e diretórios
	Alteração das características de arquivos e diretórios
	Abertura e fechamento de arquivos
	Leitura e gravação em arquivos
	Obtenção de informações sobre arquivos e diretórios
Gerência de dispositivos	Alocação e desalocação de dispositivos
	Operações de entrada/saída em dispositivos
	Obtenção de informações sobre dispositivos

Chamada a Rotinas do Sistema Operacional

- Cada **sistema operacional** possui seu **próprio conjunto de rotinas**, com nomes, parâmetros e formas de ativação específicos.
- Consequentemente, **uma aplicação desenvolvida utilizando serviços de um determinado sistema operacional não pode ser portada diretamente para um outro sistema**, exigindo algumas correções no código-fonte.

Linguagem de Comandos

- A linguagem de comandos, ou linguagem de controle, **permite que o usuário se comunique de uma forma simples com o sistema operacional, capacitando-o a executar diversas tarefas específicas do sistema** como criar, ler ou eliminar arquivos, consultar diretórios ou verificar a data e a hora armazenadas no sistema.
- Dessa forma, o usuário dispõe de uma interface direta com o sistema operacional.

Linguagem de Comandos

- **Cada sistema operacional possui a sua linguagem de comandos** como, por exemplo, a DCL (Digital Command Language) utilizada no OpenVMS, JCL (Job Control Language) no MVS da IBM e os comandos do shell disponíveis nos diversos sistemas Unix.

Exemplos de comandos do MS Windows

dir	Lista o conteúdo de um diretório
cd	Altera o diretório default
type	Exibe o conteúdo de um arquivo
del	Elimina arquivos
mkdir	Cria um diretório
ver	Mostra a versão do Windows

Linguagem de Comandos

- Cada comando, depois de digitado pelo usuário, é **interpretado pelo shell ou interpretador de comandos**, que verifica a sintaxe do comando, faz chamadas a rotinas do sistema e apresenta um resultado ou uma mensagem informativa.
- Em geral, **o interpretador de comandos não faz parte do núcleo do sistema operacional, possibilitando maior flexibilidade na criação de diferentes linguagens de controle para um mesmo sistema**.

Interface do usuário com o sistema operacional

Shell ou interpretador de comandos

Interface do usuário com o sistema operacional

- Na maioria dos sistemas operacionais, as linguagens de comandos evoluíram no sentido de **permitir a interação mais amigável com os usuários, utilizando interfaces gráficas como janelas e ícones**, a exemplo dos sistemas MS Windows.
- Na maioria dos casos, a **interface gráfica** é apenas mais um **nível de abstração entre o usuário e os serviços do sistema operacional**.

Interface do usuário com o sistema operacional

Interface gráfica

Linguagens de comandos

- As linguagens de comandos são poderosas a ponto de oferecer a **possibilidade de criar programas com estruturas de decisão e iteração**.
- Esses programas nada mais são do que uma **sequência de comandos armazenados em um arquivo texto, denominados arquivos de comandos, arquivos batch ou shell scripts, que podem ser executados sempre que necessário**.
- **Uma das principais vantagens no uso de arquivos de comandos é possibilitar a automatização de diversas tarefas ligadas à gerência do sistema**.

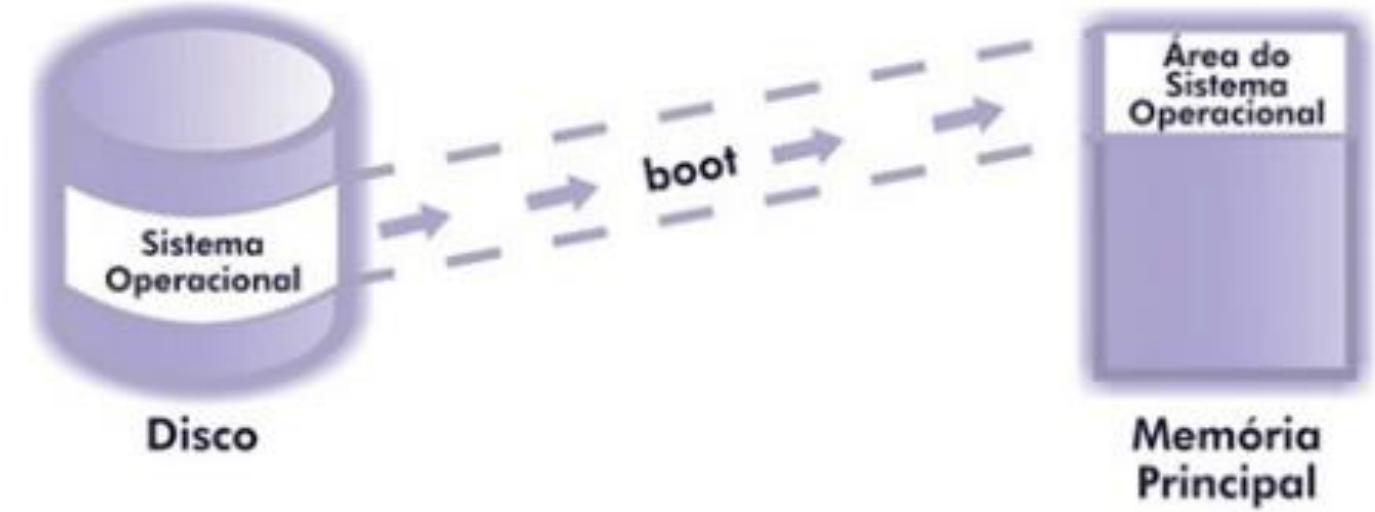
Ativação/Desativação do Sistema

- **Inicialmente, quando um computador é ligado não há sistema operacional carregado na memória da máquina.**
- **Em geral, o sistema operacional reside em um disco rígido**, podendo também estar armazenado em outros dispositivos de memória secundária, como CD ou DVD.
- **Os componentes do sistema operacional devem ser carregados para a memória principal toda vez que o computador é ligado por intermédio de um procedimento denominado ativação do sistema ou boot**.

Ativação/Desativação do Sistema

- **O procedimento de ativação se inicia com a execução de um programa chamado boot loader**, que se localiza em um endereço fixo de uma memória ROM da máquina. Este programa **chama a execução de outro programa** conhecido como **POST (Power-On Self Test)**, que **identifica possíveis problemas de hardware** no equipamento.
- **Após esta fase, o procedimento de ativação verifica se há no sistema computacional algum dispositivo de armazenamento onde haja um sistema operacional residente.**
- **Se um dispositivo com o sistema operacional é encontrado, um conjunto de instruções é carregado para memória e localizado em um bloco específico do dispositivo conhecido como setor de boot (boot sector). A partir da execução deste código, o sistema operacional é finalmente carregado para a memória principal.**
- **Além da carga, a ativação do sistema também consiste na execução de arquivos de inicialização** onde são especificados procedimentos de customização e configuração de hardware e software específicos para cada ambiente

Ativação/Desativação do Sistema



Ativação/Desativação do Sistema

- Na maioria dos sistemas também existe o processo de **desativação ou shutdown**.
- Este procedimento **permite que as aplicações e componentes do sistema operacional sejam desativados ordenadamente, garantindo, desta forma, sua integridade**.

Arquiteturas do Núcleo

- **O projeto de um sistema operacional é bastante complexo** e deve atender a diversos requisitos, algumas vezes conflitantes, como confiabilidade, portabilidade, fácil manutenção, flexibilidade e desempenho.
- **O projeto do sistema irá depender muito da arquitetura do hardware a ser utilizado e do tipo de sistema que se deseja construir**: batch, tempo compartilhado, monousuário ou multiusuário, tempo real, etc.

Arquiteturas do Núcleo

- **Os primeiros sistemas operacionais foram desenvolvidos integralmente em assembly. Com a evolução dos sistemas, técnicas de programação modular foram incorporadas ao projeto, além de linguagens de alto nível.**
- **Além de facilitar o desenvolvimento e a manutenção do sistema, a utilização de linguagens de alto nível permite uma maior portabilidade, ou seja, que o sistema operacional seja facilmente alterado em outra arquitetura de hardware**.
- **Uma desvantagem é a perda de desempenho**. Por isso, as partes críticas do sistema, como os device drivers, o escalonador e as rotinas de tratamento de interrupções, são desenvolvidas em assembly.

Arquiteturas do Núcleo

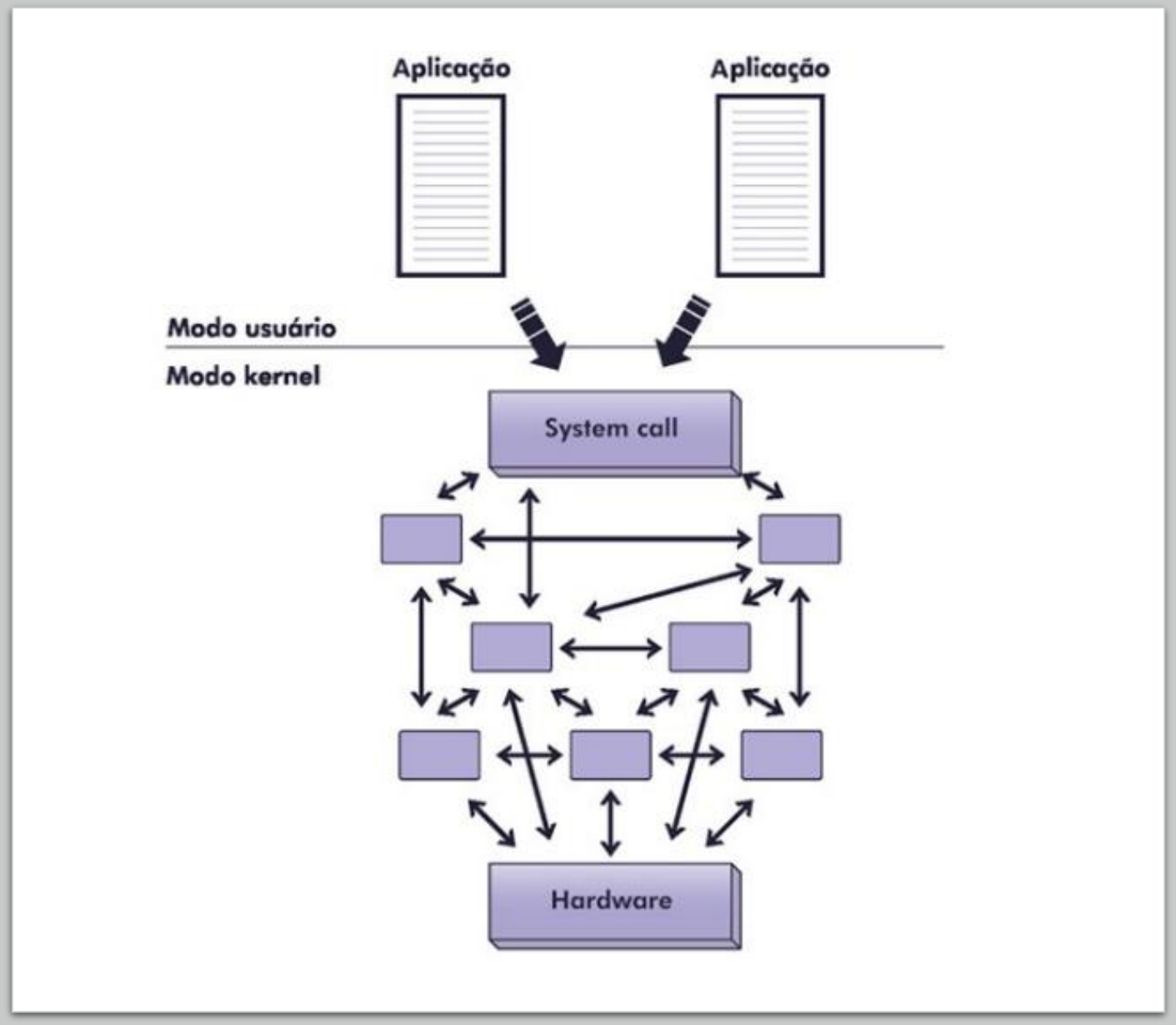
- **Uma tendência no projeto de sistemas operacionais modernos é a utilização de técnicas de orientação por objetos**, o que leva para o projeto do núcleo do sistema todas as vantagens deste modelo de desenvolvimento de software.

Algumas vantagens:

- Melhoria na organização das funções e recursos do sistema;
- Redução no tempo de desenvolvimento;
- Maior facilidade na manutenção e extensão do sistema;
- Facilidade de implementação do modelo de computação distribuída.

Arquitetura Monolítica

- A arquitetura monolítica pode ser comparada com **uma aplicação formada por vários módulos** que são compilados separadamente e depois linkados, **formando um grande e único programa executável**, onde os módulos podem interagir livremente.



Arquitetura Monolítica

- Os **primeiros sistemas operacionais foram desenvolvidos com base neste modelo**, o que tornava seu desenvolvimento e, principalmente, sua manutenção bastante difíceis.
- Devido a sua simplicidade e bom desempenho, a estrutura monolítica foi adotada no projeto do MS-DOS e nos primeiros sistemas Unix.

Arquitetura de Camadas

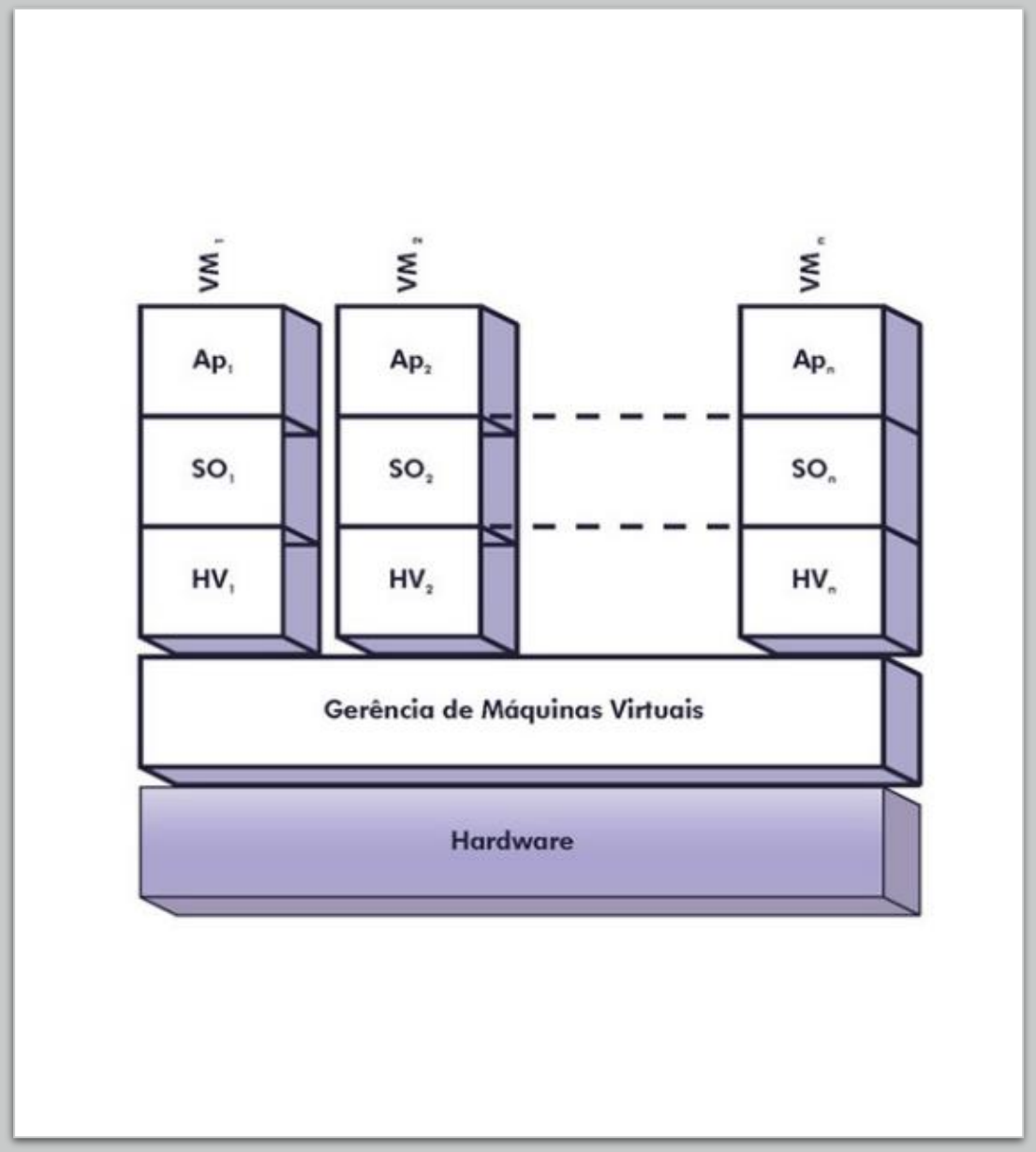
- Com o aumento da complexidade e do tamanho do código dos sistemas operacionais, técnicas de programação estruturada e modular foram incorporadas ao seu projeto. Na arquitetura de camadas**, o sistema é dividido em níveis sobrepostos. Cada camada oferece um conjunto de funções que podem ser utilizadas apenas pelas camadas superiores.**
- O primeiro sistema com base nesta abordagem foi o sistema THE (Technische Hogeschool Eindhoven), construído por Dijkstra na Holanda em 1968 e que utilizava seis camadas. Posteriormente, os sistemas MULTICS e OpenVMS também implementaram o conceito de camadas, sendo estas concêntricas. **Neste tipo de implementação, as camadas mais internas são mais privilegiadas que as mais externas.**

Arquitetura de Camadas

- **A vantagem da estruturação em camadas é isolar as funções do sistema operacional, facilitando sua manutenção e depuração, além de criar uma hierarquia de níveis de modos de acesso, protegendo as camadas mais internas. Uma desvantagem** para o modelo de camadas **é o desempenho**. Cada nova camada implica uma mudança no modo de acesso. Por exemplo, no caso do OpenVMS, para se ter acesso aos serviços oferecidos pelo kernel é preciso passar por três camadas ou três mudanças no modo de acesso.
- **Atualmente, a maioria dos sistemas comerciais utiliza o modelo de duas camadas, onde existem os modos de acesso usuário (não privilegiado) e kernel (privilegiado)**. A maioria das versões do Unix e o Windows da Microsoft está baseada neste modelo.

Máquina Virtual

- Um sistema computacional é formado por níveis, onde a camada de nível mais baixo é o hardware. Acima desta camada encontramos o sistema operacional que oferece suporte para as aplicações.
- **O modelo de máquina virtual**, ou virtual machine (VM), **cria um nível intermediário entre o hardware e o sistema operacional, denominado gerência de máquinas virtuais**. **Este nível cria diversas máquinas virtuais independentes, onde cada uma oferece uma cópia virtual do hardware, incluindo os modos de acesso, interrupções, dispositivos de E/S, etc**.



Máquina Virtual

- **Como cada máquina virtual é independente das demais, é possível que cada VM tenha seu próprio sistema operacional e que seus usuários executem suas aplicações como se todo o computador estivesse dedicado a cada um deles**.
- Na década de 1960, a IBM implementou este modelo no sistema VM/370, permitindo que aplicações batch, originadas de antigos sistemas OS/360, e aplicações de tempo compartilhado pudessem conviver na mesma máquina de forma transparente a seus usuários e aplicações.

Máquina Virtual

- **Este modelo cria o isolamento total entre cada VM, oferecendo grande segurança para cada máquina virtual**.
- **Se, por exemplo, uma VM executar uma aplicação que comprometa o funcionamento do seu sistema operacional, as demais máquinas virtuais não sofrerão qualquer problema**.
- Apesar do isolamento das aplicações, as máquinas virtuais não estão livres de problemas, como, por exemplo, vírus e erros de software do sistema operacional ou aplicações da VM.

Máquina Virtual

Além dessas vantagens, existem diversas aplicações para a utilização de máquinas virtuais:

- **Portabilidade de código**: um programa executável, em linguagem de máquina, precisa de uma plataforma de hardware específica para ser executado, tornando-o dependente desse ambiente. Se o programa for gerado em uma linguagem intermediária e uma VM fizer a tradução dos comandos para a plataforma onde o programa está sendo executado, a aplicação pode ser portada para qualquer ambiente sem a necessidade de se reescrever o programa.
- **A máquina virtual Java (Java Virtual Machine – JVM) funciona dessa forma**. O compilador Java gera um código intermediário, chamado bytecode, que é interpretado pela JVM e convertido para o ambiente onde está sendo executado. Dessa forma, o programa é desenvolvido apenas uma vez e pode ser processado em qualquer hardware ou sistema operacional, bastando que exista uma JVM para tal.

Máquina Virtual

- **Consolidação de servidores**: é muito comum que os servidores de uma empresa funcionem com apenas parte de sua capacidade total de processamento, ou seja, existe a subutilização da UCP, memória principal e dispositivos de E/S dos sistemas computacionais.
- **A ideia da consolidação é poder executar diversas aplicações que estariam sendo processadas em diferentes servidores, cada qual como seu hardware e sistema operacional, em uma única máquina, utilizando uma VM para cada uma das aplicações**. Dessa forma, a consolidação permite a melhor utilização dos recursos computacionais e, além disso, também permite a redução de custos de infraestrutura, como energia elétrica, refrigeração, espaço físico e cabeamento de rede.

Máquina Virtual

- **Aumento da disponibilidade**: uma vez que cada VM pode ser copiada facilmente para a memória secundária, como um disco, caso haja algum problema com uma VM ou com o próprio sistema onde está sendo processada, basta restaurar a cópia da VM em outro servidor e rapidamente o ambiente pode ser novamente disponibilizado.

Máquina Virtual

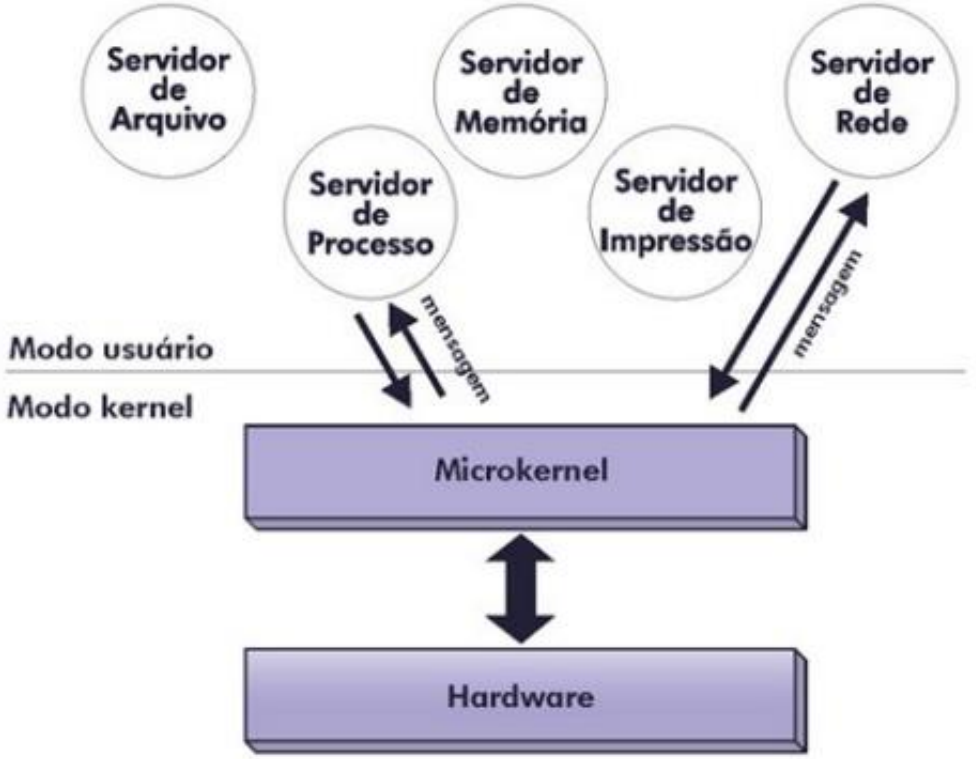
- **Facilidade de escalabilidade e balanceamento de carga**: caso o sistema no qual a VM está sendo processada fique sobrecarregado e apresente problemas de desempenho, a VM pode ser facilmente migrada para um novo ambiente que possua maior capacidade de processamento. Dessa forma, é possível balancear a carga dos servidores e garantir o bom desempenho das aplicações.
- **Facilidade no desenvolvimento de software**: as VMs permitem a criação de um ambiente independente para o desenvolvimento e teste de software sem o comprometimento das máquinas de produção. Além disso, é possível o teste do software em diferentes sistemas operacionais e suas versões sem a necessidade de um hardware dedicado.

Arquitetura Microkernel

- **Uma tendência nos sistemas operacionais modernos é tornar o núcleo do sistema operacional o menor e mais simples possível**.
- Para implementar esta ideia, **os serviços do sistema são disponibilizados através de processos, onde cada um é responsável por oferecer um conjunto específico de funções, como gerência de arquivos, gerência de processos, gerência de memória e escalonamento**.

Arquitetura Microkernel

- **Sempre que uma aplicação deseja algum serviço, é realizada uma solicitação ao processo responsável**.
- A aplicação que solicita o serviço é chamada de cliente, enquanto o processo que responde à solicitação é chamado de servidor.
- Um cliente, que pode ser uma aplicação de um usuário ou um outro componente do sistema operacional, solicita um serviço enviando uma mensagem para o servidor. O servidor responde ao cliente através de uma outra mensagem.
- **A principal função do núcleo é realizar a comunicação, ou seja, a troca de mensagens entre cliente e servidor.**



Arquitetura Microkernel

O conceito da arquitetura microkernel surgiu no sistema operacional Mach, na década de 1980, na Universidade Carnegie-Mellon. O núcleo do sistema Mach oferece basicamente quatro serviços: gerência de processos, gerência de memória, comunicação por troca de mensagens e operações de E/S, todos em modo usuário.

A utilização deste modelo permite que os servidores executem em modo usuário, ou seja, não tenham acesso direto a certos componentes do sistema. **Apenas o núcleo do sistema, responsável pela comunicação entre clientes e servidores, executa no modo kernel.** Como consequência, se ocorrer um erro em um servidor, este poderá parar, mas o sistema não ficará inteiramente comprometido, aumentando assim a sua disponibilidade.

Arquitetura Microkernel

- Como os servidores se comunicam através de trocas de mensagens, não importa se os clientes e servidores são processados em um sistema com um único processador, com múltiplos processadores (fortemente acoplado) ou ainda em um ambiente de sistema distribuído (fracamente acoplado).
- **A implementação de sistemas microkernel em ambientes distribuídos permite que um cliente solicite um serviço e a resposta seja processada remotamente. Esta característica permite acrescentar novos servidores à medida que o número de clientes aumenta, conferindo uma grande escalabilidade ao sistema operacional.**

Arquitetura Microkernel

- **A arquitetura microkernel permite isolar as funções do sistema operacional** por diversos processos servidores pequenos e dedicados a serviços específicos**, tornando o núcleo menor, mais fácil de depurar e, consequentemente, aumentando sua confiabilidade**.
- **Na arquitetura microkernel, o sistema operacional passa a ser de mais fácil manutenção, flexível e de maior portabilidade.**

Arquitetura Microkernel

- Apesar de todas as vantagens deste modelo, **sua implementação, na prática, é muito difícil.**
- Primeiro **existe o problema de desempenho, devido à necessidade de mudança de modo de acesso a cada comunicação entre clientes e servidores**. **Outro problema é que certas funções do sistema operacional exigem acesso direto ao hardware**, como operações de E/S.
- **Na realidade, o que é implementado mais usualmente é uma combinação do modelo de camadas com a arquitetura microkernel. O núcleo do sistema, além de ser responsável pela comunicação entre cliente e servidor, passa a incorporar outras funções críticas do sistema**, como escalonamento, tratamento de interrupções e gerência de dispositivos.

Arquitetura Microkernel

- **Existem vários projetos baseados em sistemas microkernel, principalmente em instituições de ensino e centros de pesquisa**, como o Exokernel, do MIT (Massachusetts Institute of Technology), L4, da Universidade de Dresden, e Amoeba, da Vrije Universiteit.
- **A maioria das iniciativas** nesta área está **relacionada ao desenvolvimento de sistemas operacionais distribuídos**.