SISTEMAS OPERACIONAIS

AULA 4 – CONCEITOS AVANÇADOS

Prof.ª Sandra Cossul, Ma.

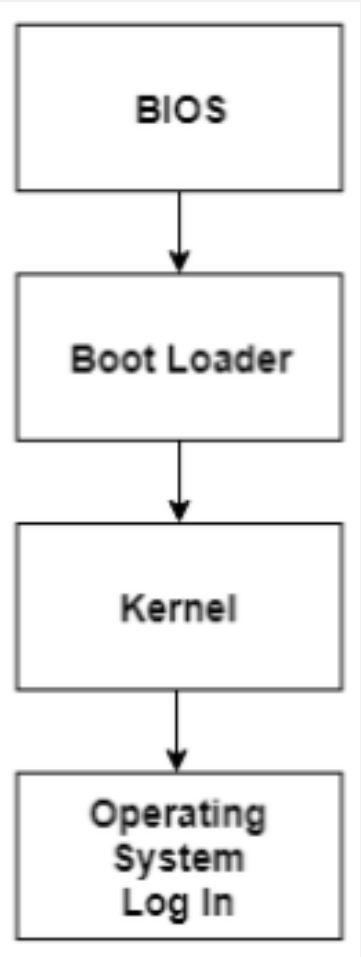
TÓPICOS DA AULA

- Etapas de inicialização do SO
- Chamadas de sistema
- Estrutura dos SOs

ETAPAS DE INICIALIZAÇÃO DO SO

ETAPAS DE INICIALIZAÇÃO DO SO (BOOT)

1. CPU inicializa ao ligar o computador
2. CPU procura a **ROM BIOS** do sistema para obter a **primeira instrução** no programa de inicialização que instrui o sistema a executar o POST
   - **POST** (Power on Self Test) - conjunto inicial de testes de diagnóstico realizados pelo computador logo após ele ser ligado, com a intenção de verificar se há problemas relacionados ao hardware.
3. Pequeno pedaço de código conhecido como **programa bootstrap ou boot loader** localiza o **kernel** (encontra a unidade de inicialização apropriada do SO)

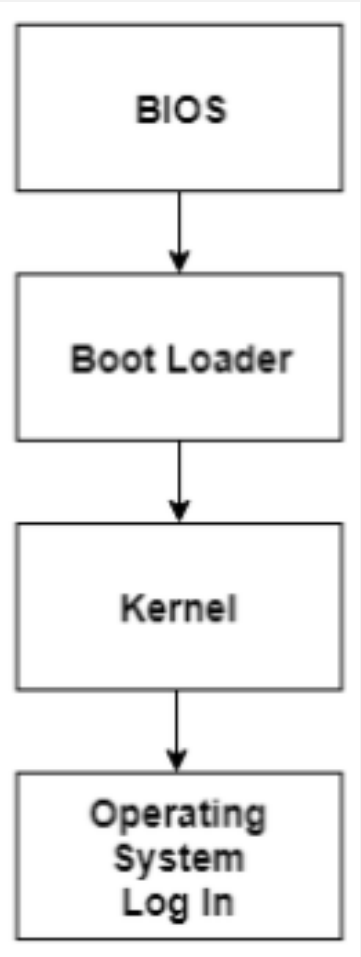


ETAPAS DE INICIALIZAÇÃO DO SO (BOOT)

4. BIOS copia os arquivos do SO (kernel) para a **memória**. Em seguida, o SO controla o **processo de inicialização.**

5. O kernel **inicializa o hardware** (carrega os drivers de dispositivo necessários para controlar os dispositivos periféricos)

6. Os **usuários** podem acessar os **aplicativos do sistema** para realizar várias tarefas.



RELEMBRANDO...MODOS DE OPERAÇÃO

- Níveis básicos de privilégio:
  - **Modo núcleo (kernel)**
    - todas funcionalidades do CPU estão disponíveis: todos os recursos internos (registradores e portas de E/S) e áreas de memória podem ser acessados.
  - **Modo usuário**
    - somente um subconjunto das instruções do CPU e registradores estão disponíveis
    - instruções como RESET e IN/OUT são proibidas

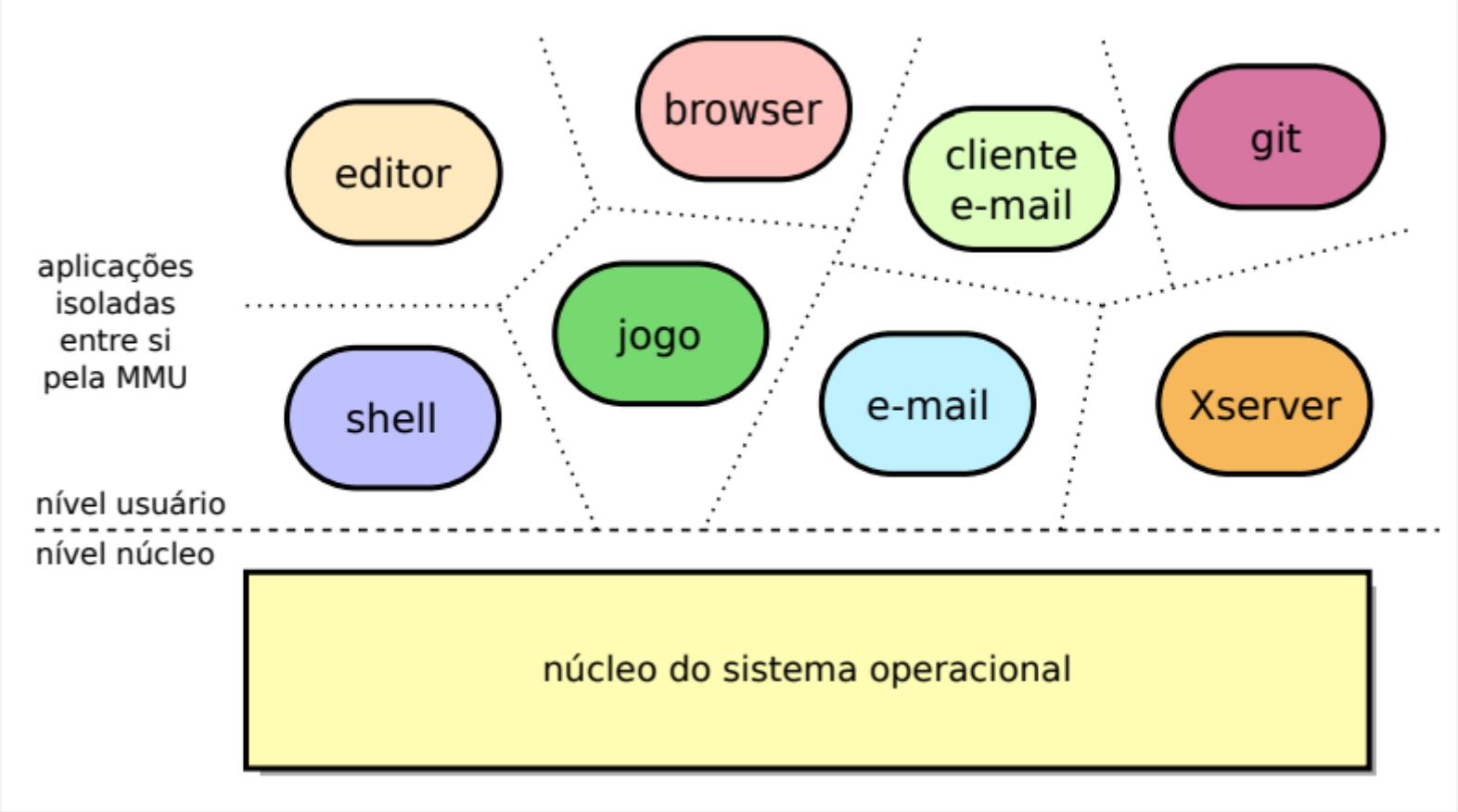
RELEMBRANDO...MODOS DE OPERAÇÃO

- O núcleo do SO, bem como drivers e código de inicialização executam em **modo núcleo**, enquanto os programas utilitários e os aplicativos executam em **modo usuário**

- Os **flags** que definem o **nível de privilégio** só podem ser **modificados por código executando no nível núcleo**, o que impede usuários maliciosos de contornar essa barreira de proteção

- Além dos níveis de privilégio, o núcleo do SO pode configurar a unidade de gerência de memória (MMU) para criar uma área de memória exclusiva para cada aplicação (isolada das demais aplicações e do núcleo).

RELEMBRANDO...MODOS DE OPERAÇÃO



SYSTEM CALLS (CHAMADAS DE SISTEMA)

CHAMADAS DE SISTEMA

- A proteção introduz um problema:
  - **Como invocar, a partir da aplicação, as rotinas oferecidas pelo núcleo para o acesso ao hardware e demais serviços do SO?**

Mecanismo de interrupção de software (trap)

A execução é desviada para um endereço predefinido (desvio incondicional) e o CPU passsa a operar em nível núcleo (elevação de privilégio)

A ativação de uma rotina do núcleo usando esse mecanismo é denominada **chamada de sistema.**

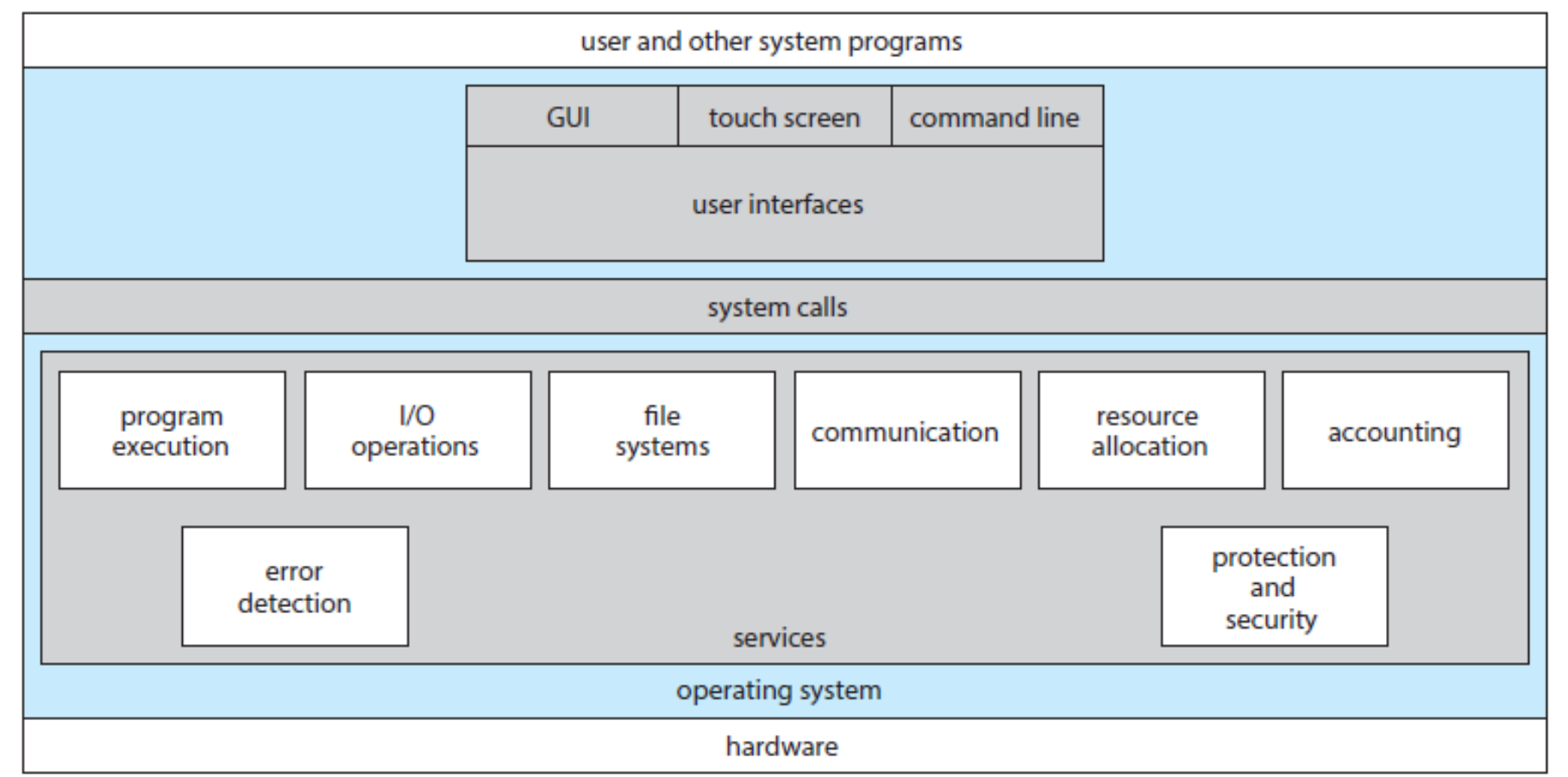
CHAMADAS DE SISTEMA

É a maneira programática com que um programa requisita um serviço do sistema operacional.

- Fornecem uma **interface** para os **serviços disponibilizados** por um sistema operacional (forma de interação).
- Tipicamente escritas em linguagem de alto nível (C ou C++)

- São funções especiais que gerenciam as rotinas do sistema operacional que ocorrem no **modo kernel.**

CHAMADAS DE SISTEMA

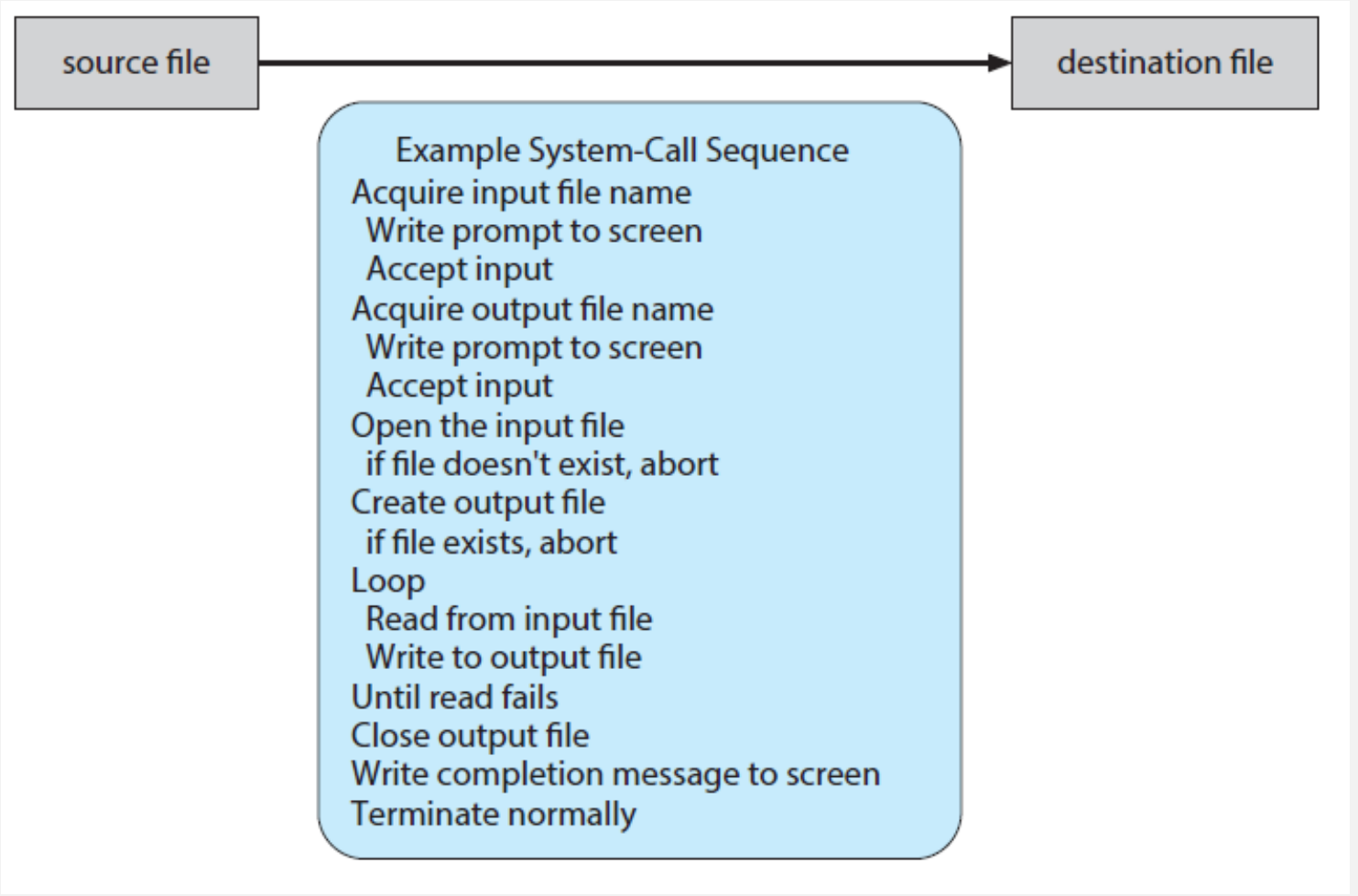


CHAMADAS DE SISTEMA

- Os sistemas operacionais definem chamadas de sistema para todas as operações envolvendo o **acesso a recursos de baixo nível** (periféricos, arquivos, alocação de memória, etc.) ou **abstrações lógicas** (criação e encerramento de tarefas, operadores de sincronização, etc.)

CHAMADAS DE SISTEMA

- **Ex.:** Copiar dados de um arquivo para outro



CHAMADAS DE SISTEMA

- As chamadas de sistema fornecem os serviços do SO para os programas de usuário por meio da **API (Application Programming Interface)** do SO
  - Especifica as funções disponíveis para um programador
  - *Win32* (Microsoft) e *API POSIX* (sistemas UNIX)

- **Porque não utilizar diretamente chamadas de sistema?**
  - Portabilidade
  - Facilidade de implementação (bibliotecas, funções prontas)
  - RTE

CHAMADAS DE SISTEMA

- **RTE (run-time environment)**
  - Conjunto completo de software necessário para executar aplicações escritas em determinada linguagem de programação, incluindo seus compiladores ou interpretadores.

- **RTE fornece uma interface para chamadas de sistema**
  - Intercepta chamadas de funções da API e inicia as chamadas de sistema necessários dentro do SO
  - Detalhes da interface do SO estão ocultos do programador devido a API e são gerenciados pelo RTE

TIPOS DE CHAMADAS DE SISTEMA

- **Controle e gestão de processos**
  - Criar e finalizar processo
  - Carregar, executar
  - Obter e definir atributos de processo
  - Alocar e liberar memória
  - Eventos de espera
- **Gerenciamento de arquivos**
  - Criar e excluir arquivo
  - Abrir, fechar
  - Ler, escrever, atualizar
  - Obter e setar atributos de arquivos

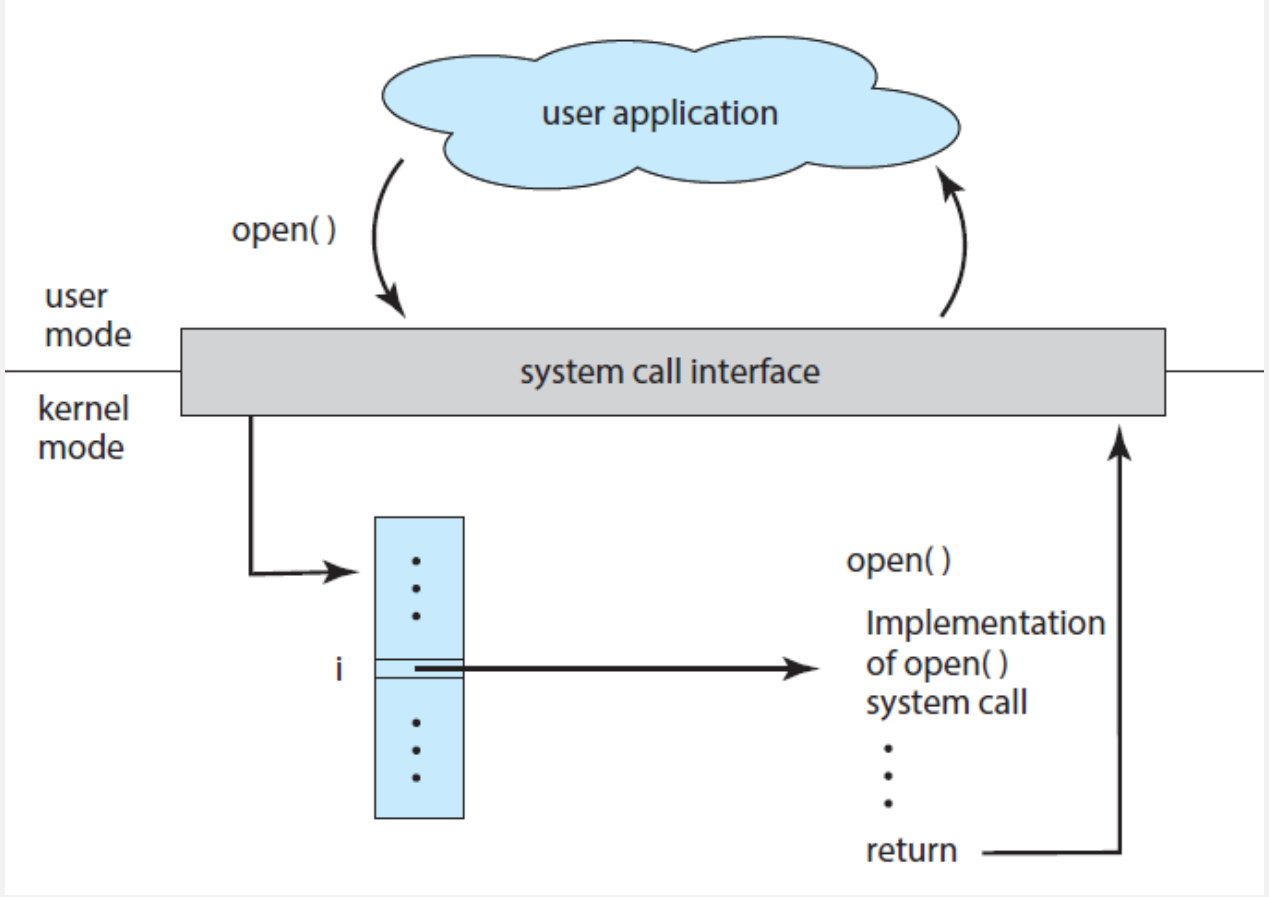
TIPOS DE CHAMADAS DE SISTEMA

- **Gerenciamento de dispositivos**
  - Solicitar e liberar dispositivo
  - Ler, escrever e atualizar
  - Obter e definir atributos
  - Conectar e desconectar dispositivos logicamente
- **Manutenção de informações**
  - Obter e definir hora e data
  - Obter e definir dados do sistema
  - Obter e definir atributos de processos, arquivos ou dispositivos

TIPOS DE CHAMADAS DE SISTEMA

- **Gestão da memória**
  - Alocar/liberar/modificar áreas de memória
- **Comunicação**
  - Iniciar e finalizar conexão para comunicação
  - Enviar e receber mensagem
  - Transferir informações de status
  - Conectar ou desconectar dispositivos remotos
- **Proteção**
  - Obter e definir proteção de arquivos

CHAMADAS DE SISTEMA

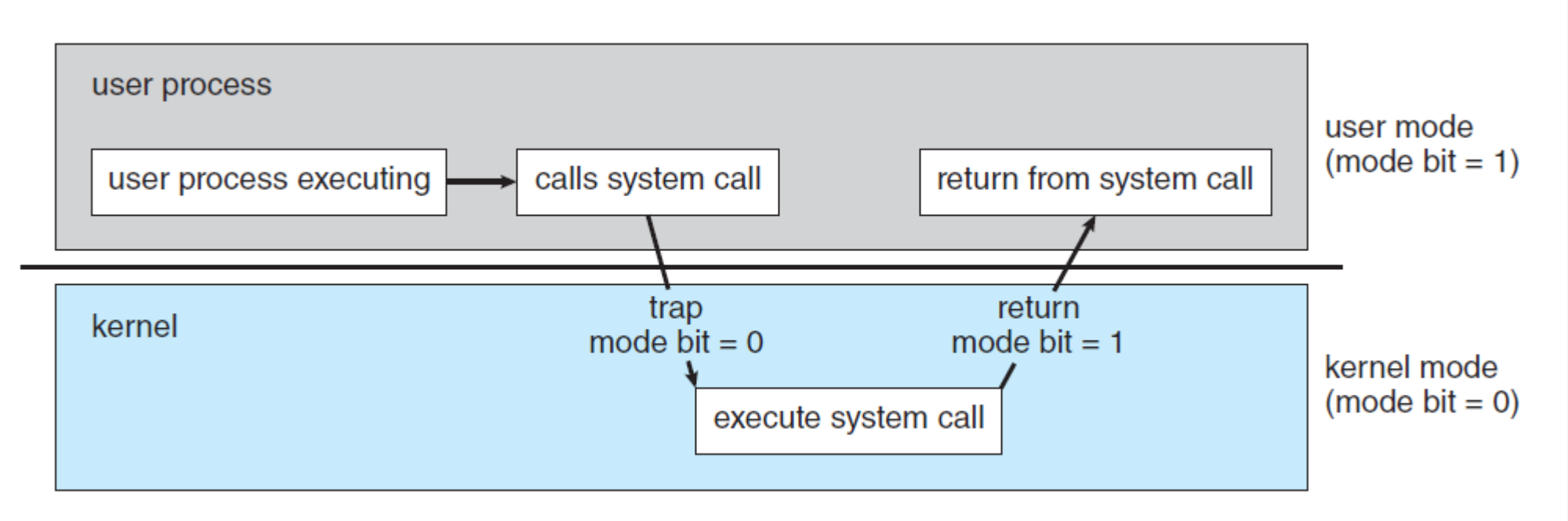


CHAMADAS DE SISTEMA

- Se um processo do usuário necessita de um **serviço do SO**, uma **chamada de sistema** é usada para solicitar a realização desta tarefa.
  - Isso exige a passagem do processador do modo usuário para o modo kernel
  - A alteração do modo é realizado dentro da rotina de system call, pela colocação de uma instrução chamada **trap**

- Uma **trap** é uma instrução que causa uma **interrupção**, mudando o modo do processador e desviando o processamento para uma rotina específica do SO

CHAMADAS DE SISTEMA



A aplicação pede um serviço ao SO (System Call – chamada de sistema).

ESTRUTURAS DE UM SISTEMA OPERACIONAL

ESTRUTURAS DE UM SISTEMA OPERACIONAL

- Sistemas grandes e complexos como um sistema operacional precisam ser **estruturados** de forma cuidadosa para funcionar corretamente além de serem fáceis de modificar caso for necessário.

- Vamos ver algumas possibilidades de **design de SOs** que foram testados na prática.

ESTRUTURAS DE UM SISTEMA OPERACIONAL

1. **Estrutura Simples/Monolíticos**
2. **Abordagem em camadas**
3. **Microkernel (micronúcleo)**
4. **Abordagem em módulos**
5. **Sistemas híbridos**

ESTRUTURAS DE UM SISTEMA OPERACIONAL

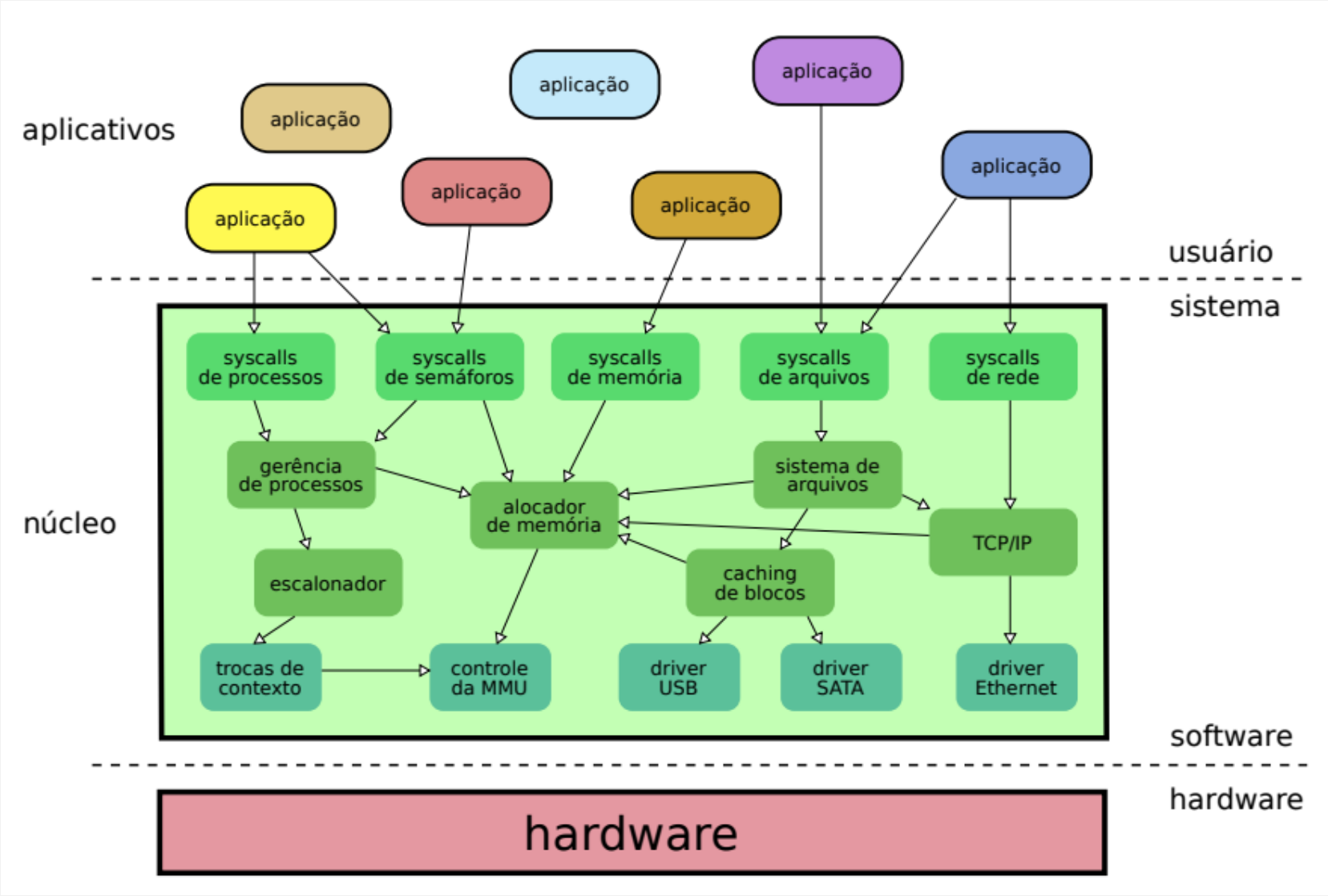
1. Estrutura Simples/Monolítico

- não tem estrutura definida
- Interfaces e níveis de funcionalidade não estão bem separados
- SO escrito como um conjunto de procedimentos onde cada um pode chamar todos os demais sem hierarquia
- **Ex.:** MS-DOS, UNIX original, FreeBSD (núcleo monolítico)
- **Vantagens:** velocidade e eficiência (comunicação com o kernel é rápido e não há sobrecarga na interface de chamadas do sistema) – interação direta
- **Problemas**: vulnerabilidade

ESTRUTURAS DE UM SISTEMA OPERACIONAL

1. Estrutura Simples/Monolítico

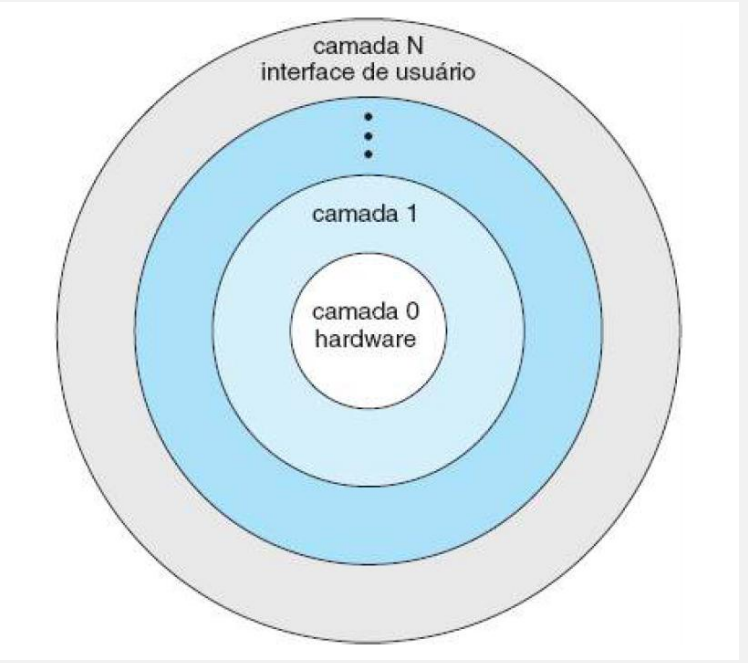
- Acesso a todos os recursos do hardwae e sem restrições de acesso à memória.



ESTRUTURAS DE UM SISTEMA OPERACIONAL

2. Abordagem em camadas

- SO dividido em vários **níveis** (camada inferior é o hardware e a camada mais externa é a interface de usuário)
- **Vantagem:** simplicidade de construção e debugging (uma camada superior não precisa se preocupar com a funcionalidade das camadas inferiores)
- **Desvantagem:** definir as funcionalidades de cada camada; pouco eficiente (overhead)



- Poucos SOs utilizam uma abordagem em camadas pura
- É utilizada a "ideia" de camadas, de forma mais simples

ESTRUTURAS DE UM SISTEMA OPERACIONAL

2. Abordagem em camadas

- **THE** – Technische Hogeschool Eindhoven (Universidade de Tecnologia de Eindhoven)
  - primeiro SO organizado nessa estrutura, desenvolvido por Edsger Dijkstra

Camada	Função
5	Controle geral do sistema (chamado de *Operador*)
4	Processos do usuário
3	Gerência de Entrada/Saída
2	Comunicação entre os processos
1	Gerência de memória e do Disco
0	Escalonamento dos processos (multiprogramação)

ESTRUTURAS DE UM SISTEMA OPERACIONAL

2. Abordagem em camadas

- Limitações para aplicação deste modelo:
  - **Empilhamento de várias camadas de software -** cada pedido de uma aplicação demore mais tempo para chegar até o dispositivo periférico ou recurso a ser acessado, prejudicando o desempenho.
  - **Divisão de funcionalidades do sistema não é óbvia –** muitas funcionalidades são interdependentes o que pode gerar conflitos

- Organização parcial em camadas: Minix 3, Windows 2000 e Android.
- HAL – *Hardware Abstraction Layer*

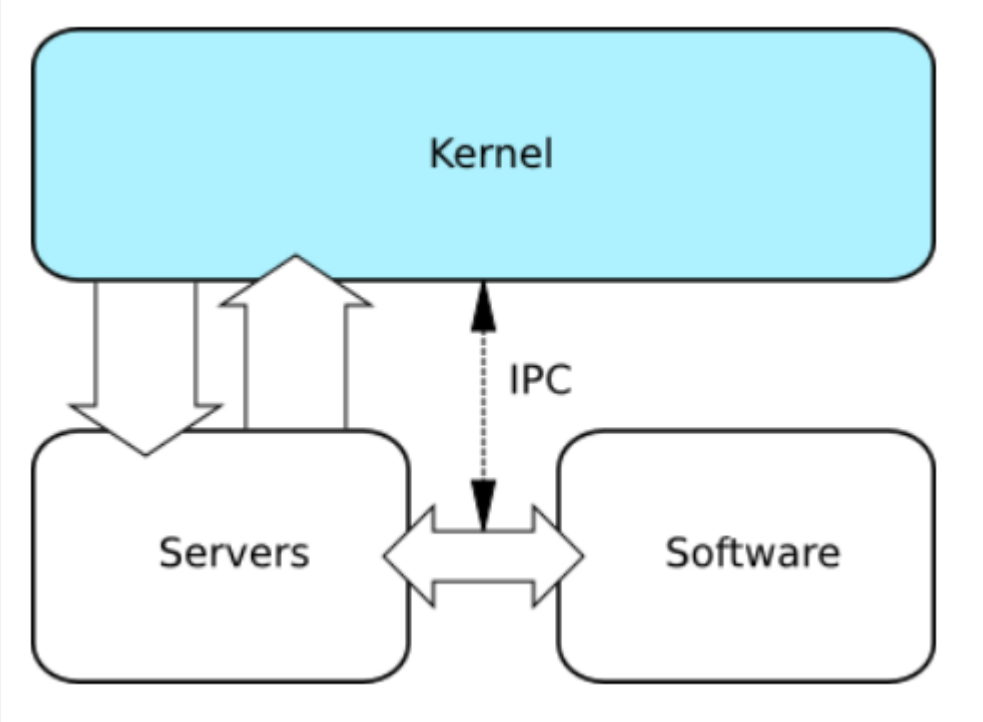
ESTRUTURAS DE UM SISTEMA OPERACIONAL

3. Microkernel (ou micronúcleo)

- Esse método estrutura o SO removendo todos os componentes não essenciais do kernel e implementando-os como programas de nível de sistema e de usuário
- A ideia básica é atingir alta confiabilidade separando o sistema operacional em pequenos e bem definidos módulos em que apenas um (o microkernel) roda em **modo kernel** e os outros módulos rodam no **modo usuário.**
- Principal função do microkernel: comunicar entre os programas e os vários serviços que também estão rodando no nível de usuário
- Resultado: Kernel menor

ESTRUTURAS DE UM SISTEMA OPERACIONAL

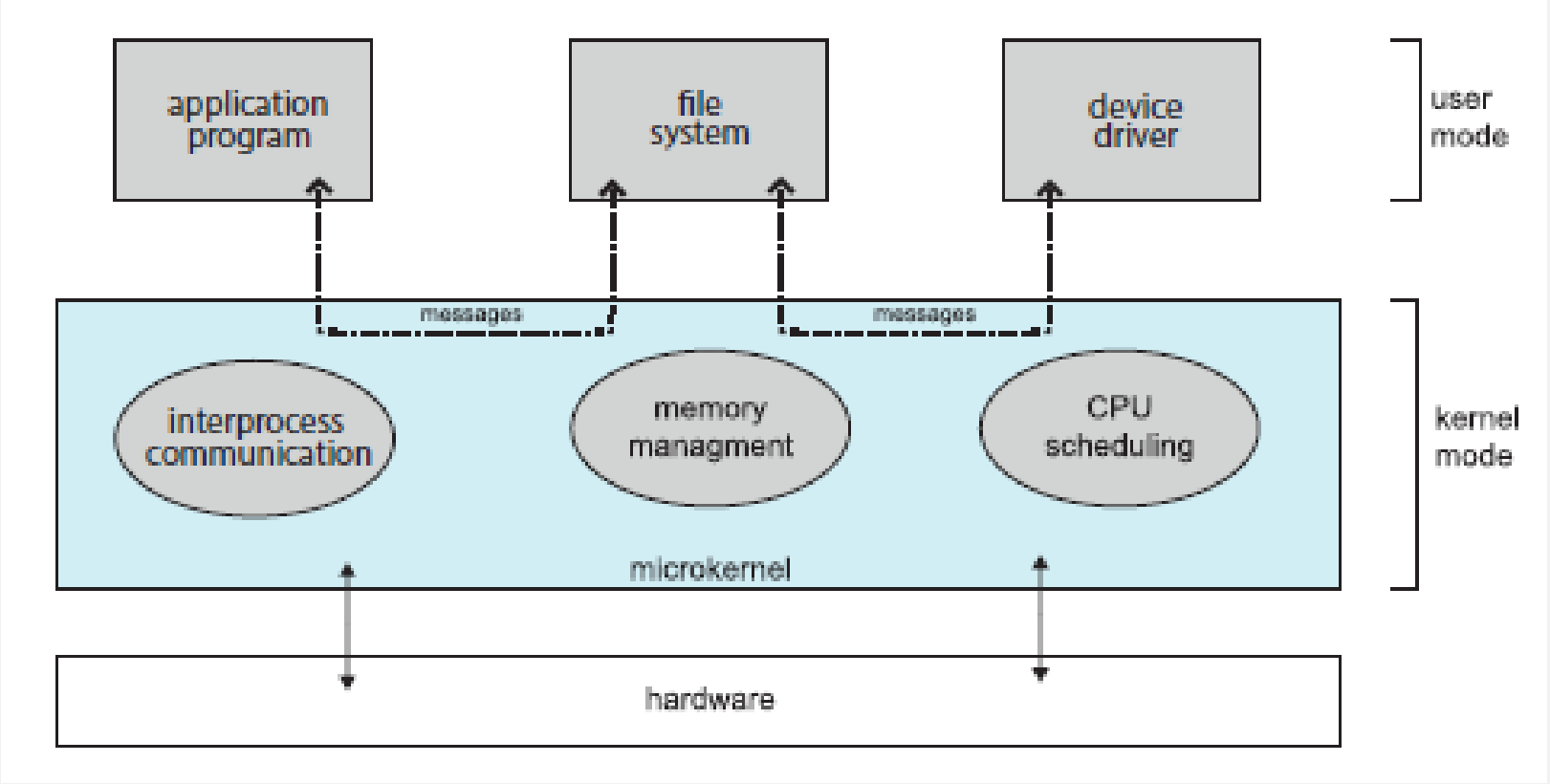
3. Microkernel (ou micronúcleo)



Ideia: retirar do núcleo todo o código de alto nível (abstrações de recursos) deixando no núcleo somente o código de baixo nível necessário para interagir com o hardware e criar abstrações básicas.

ESTRUTURAS DE UM SISTEMA OPERACIONAL

3. Microkernel



ESTRUTURAS DE UM SISTEMA OPERACIONAL

3. Microkernel

- **Vantagens:**
  - facilita a extensão do SO (todos os novos serviços são adicionados no espaço do usuário e portanto, não requer modificação do kernel)
  - segurança e confiabilidade (serviços executados como processos de usuário e não do kernel; se um serviço falha o resto do SO permanece intocado)
- **Desvantagens**
  - Aumento do *overhead* de funções de sistema

ESTRUTURAS DE UM SISTEMA OPERACIONAL

3. Microkernel

- Minix 3 (exemplo de micronúcleo bem sucedido)
- SOs que adotam parcialmente essa estrutura, adotando núcleos híbridos: MacOS da Apple, Digital UNIX e Windows NT

ESTRUTURAS DE UM SISTEMA OPERACIONAL

4. Abordagem em módulos

- **Módulos de kernel carregáveis**
- O Kernel tem um conjunto de componentes principais e vincula **serviços adicionais** por meio de módulos
  - Adicionar suporte para novo hardware e/ou sistemas de arquivos
  - Adicionar chamadas de sistema

- Serviços **implementados dinamicamente** quando o kernel está em execução
- Quando a funcionalidade fornecida por um módulo de kernel não é mais necessária, ele pode ser desvinculado, liberando memória e outros recursos.

ESTRUTURAS DE UM SISTEMA OPERACIONAL

4. Abordagem em módulos

- Parecido com um sistema em camadas (cada seção do kernel tem interfaces definidas porém é mais flexível porque um módulo pode chamar qualquer outro módulo)
- Parecido com a abordagem microkernel (módulo principal tem apenas funções específicas e o conhecimento de como carregar e se comunicar com outros módulos, mas é mais eficiente pois os módulos não precisar fazer transmissão de mensagens)

- Implementação comum em SOs modernos (Unix, Solarix, Linux, MacOS e Windows)

ESTRUTURAS DE UM SISTEMA OPERACIONAL

5. Sistemas híbridos

- Na prática, os SOs não adotam uma única estrutura!

- O que acontece é uma **combinação** das diferentes **estruturas**, resultando em sistemas híbridos que resolvem problemas de **desempenho**, **segurança** e **usabilidade**.

- Abordagem intermediária entre o núcleo monolítico e o micronúcleo.
- É comum observar também nos núcleos híbridos uma influência da arquitetura em camadas.
- Ex.: a partir da versão 4 do Windows NT, MacOS e iOS

PRÓXIMA AULA

- Interrupções
- Processos

BIBLIOGRAFIA

- Tanenbaum, A. S. **Sistemas Operacionais Modernos.** Pearson Prentice Hall. 3rd Ed., 2009.
- Silberschatz, A; Galvin, P. B.; Gagne G.; **Fundamentos de Sistemas Operacionais.** LTC. 9th Ed., 2015.
- Stallings, W.; **Operating Systems: Internals and Design Principles**. Prentice Hall. 5th Ed., 2005.