SISTEMAS OPERACIONAIS

Kernel - EAD

Análise e Desenvolvimento de Sistemas

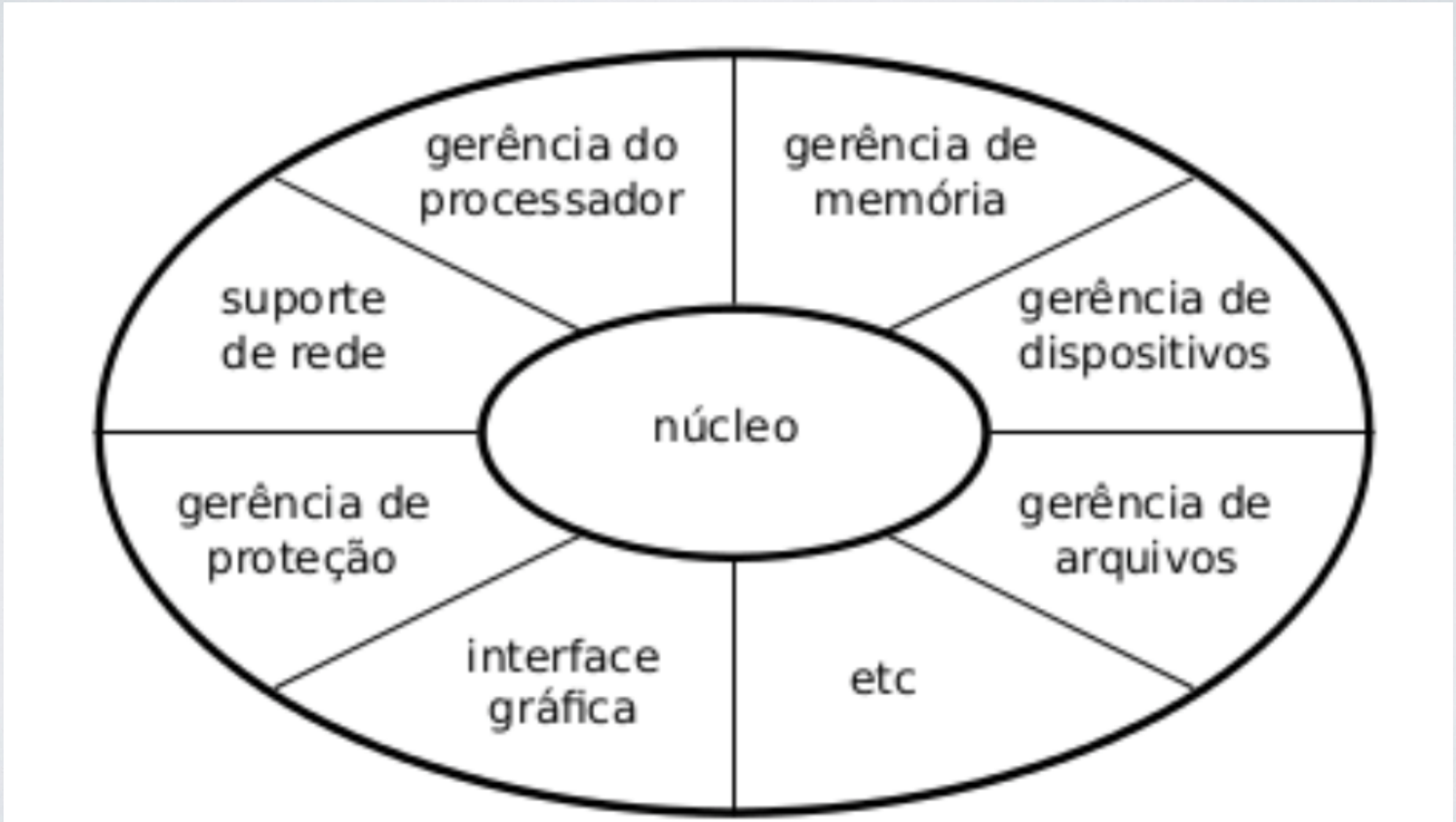
Prof. Ms. Renato Leite

renato@sectras.edu.br

OBJETIVOS

- Entender o que é o núcleo / Kernel do sistema operacional;
- Como se dividem as gerências primárias e secundárias ligadas ao Kernel;
- Como se promove a comunicação entre aplicações e o Kernel;
- Os tipos de arquitetura aplicadas aos Kernel;

S.O. - FUNÇÕES


gerência do processador
gerência de memória
suporte de rede
gerência de dispositivos
núcleo
gerência de proteção
gerência de arquivos
interface gráfica
etc

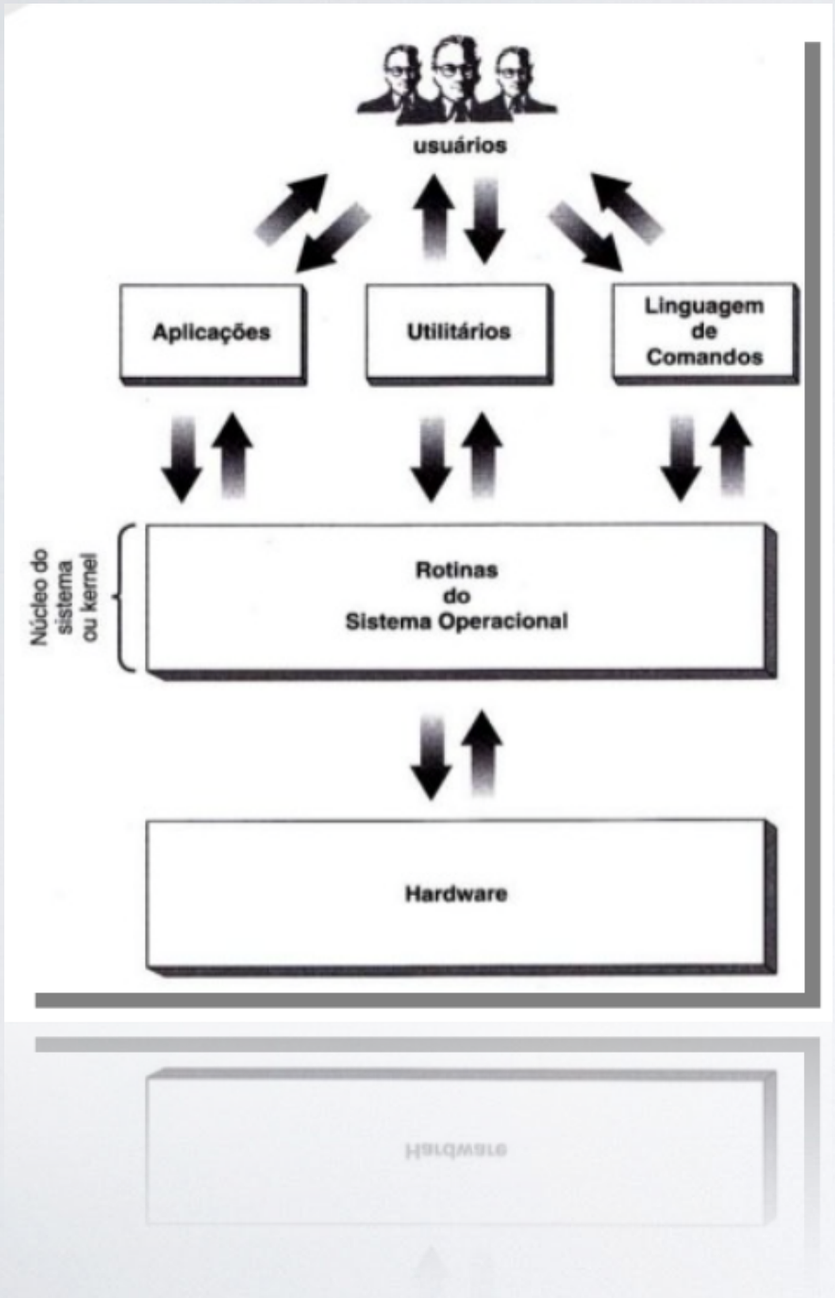
O NÚCLEO DO SISTEMA OPERACIONAL

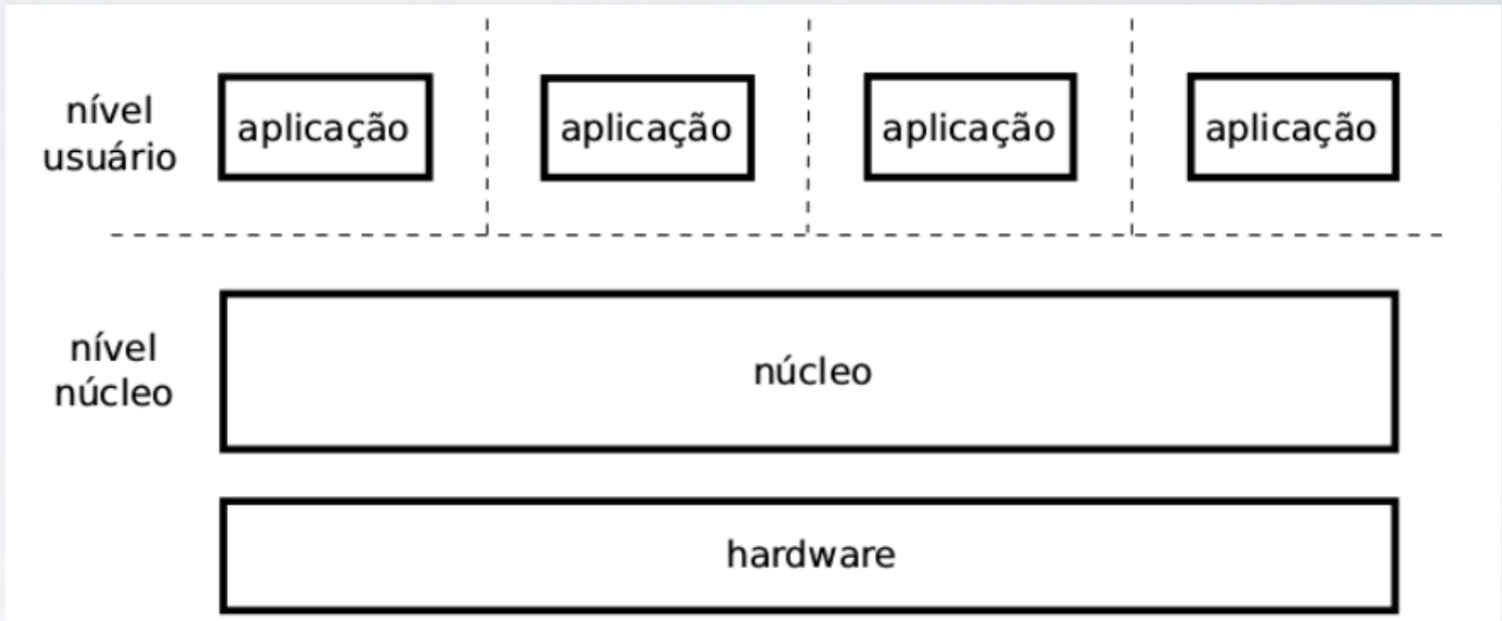
- Núcleo é um termo que designa a parte mais importante e determinante de uma estrutura. É assim em uma célula biológica, em uma empresa e não é diferente em sistemas operacionais;

- Um S.O fornece diversos serviços, tais quais acesso a memória, escalonamento de processos, gerência de arquivos, suporte a redes, acesso a dispositivos de E/S…

- Todos esses serviços são fornecidos pelo núcleo do sistema operacional, independente de sua estrutura.

KERNEL S.O


usuários
Aplicações
Utilitários
Linguagem de Comandos
Núcleo do sistema ou kernel
Rotinas do Sistema Operacional
Hardware


nível usuário
aplicação
aplicação
aplicação
aplicação
nível núcleo
núcleo
hardware

PROTEÇÃO KERNEL

- Proteger a integridade do kernel é proteger todo o sistema;
- Aplicações de usuário (fora do núcleo) só executam instruções que não possam gerar inconsistências ao sistema (modo usuário);
- Outras instruções diretas são executadas pelo núcleo em outro nível de execução (modo kernel). Ex.: marcar uma área de memória como livre ou **reset** no processador;

O NÚCLEO DO SISTEMA OPERACIONAL

- Vimos que as aplicações (desenvolvidas por terceiros ou utilitários do sistema - cp, mv, rm, ls) utilizam serviços do S.O. por não saberem acessar diretamente o hardware;

- Como o sistema fornece esses serviços?

- System calls: chamadas de sistema que indicam ao kernel que a aplicação deseja alguma recurso que não tem acesso com seu conjunto de instruções a nível de usuário;

SYSTEM CALLS

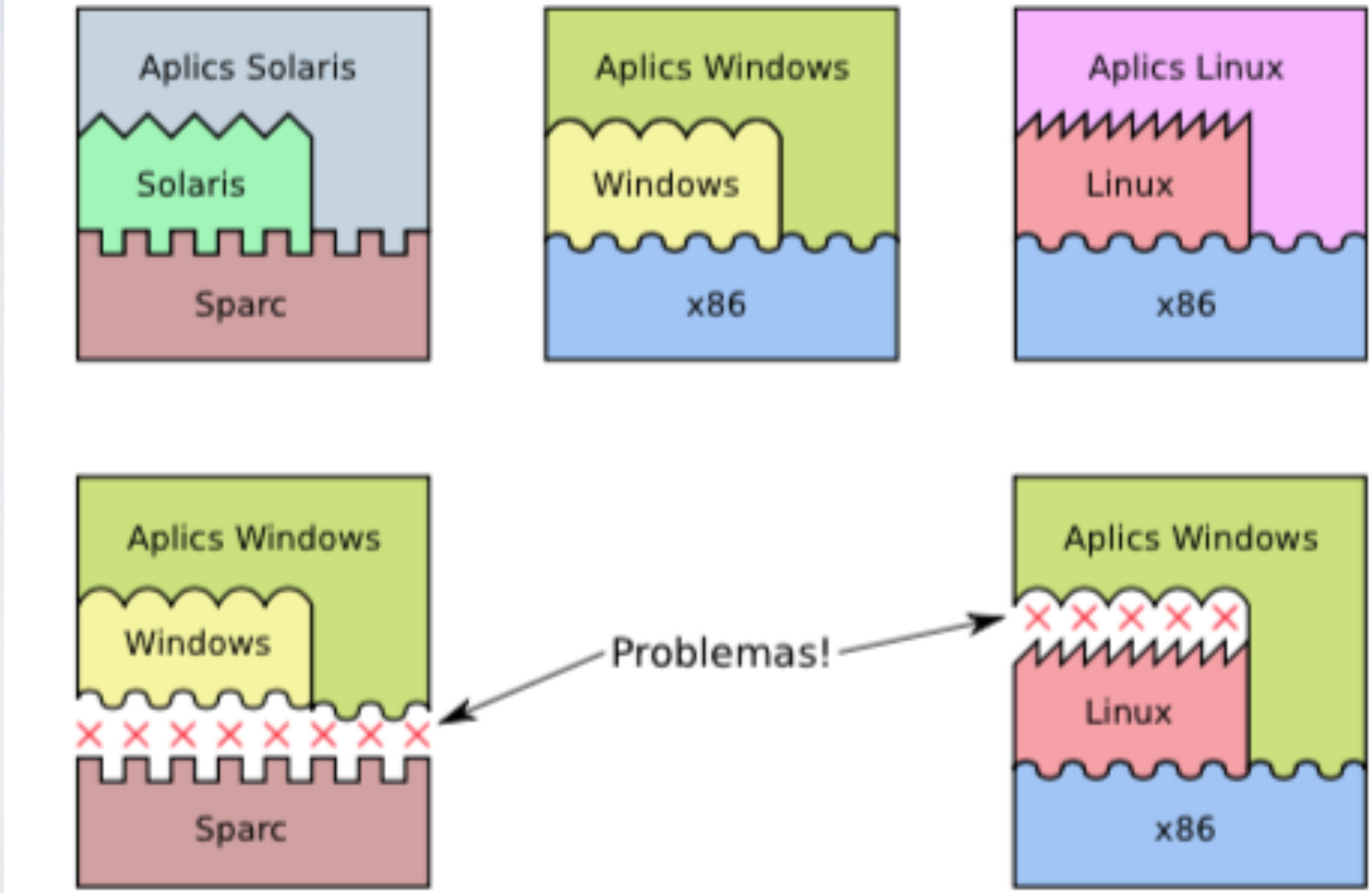
- System call é um termo muito utilizado para sistemas Unix-like, porém é um conceito que existe em outros S.O.;
- No Windows, por exemplo, chamamos de API (Application Program Interface);
- Existe um padrão POSIX que procura padronizar as chamadas de sistema Unix entre todas as derivações, facilitando a portabilidade de apps entre distribuições.

```
GetSystemTime(SystemTime);
DataHoraT := SystemTimeToDateTime(SystemTime);
DataHoraS := DateTimeToStr(DataHoraT);
RichEdit1.Lines.Add(DataHoraS);
```

SYSTEM CALLS

Funções	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; Abrir e fechar 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.

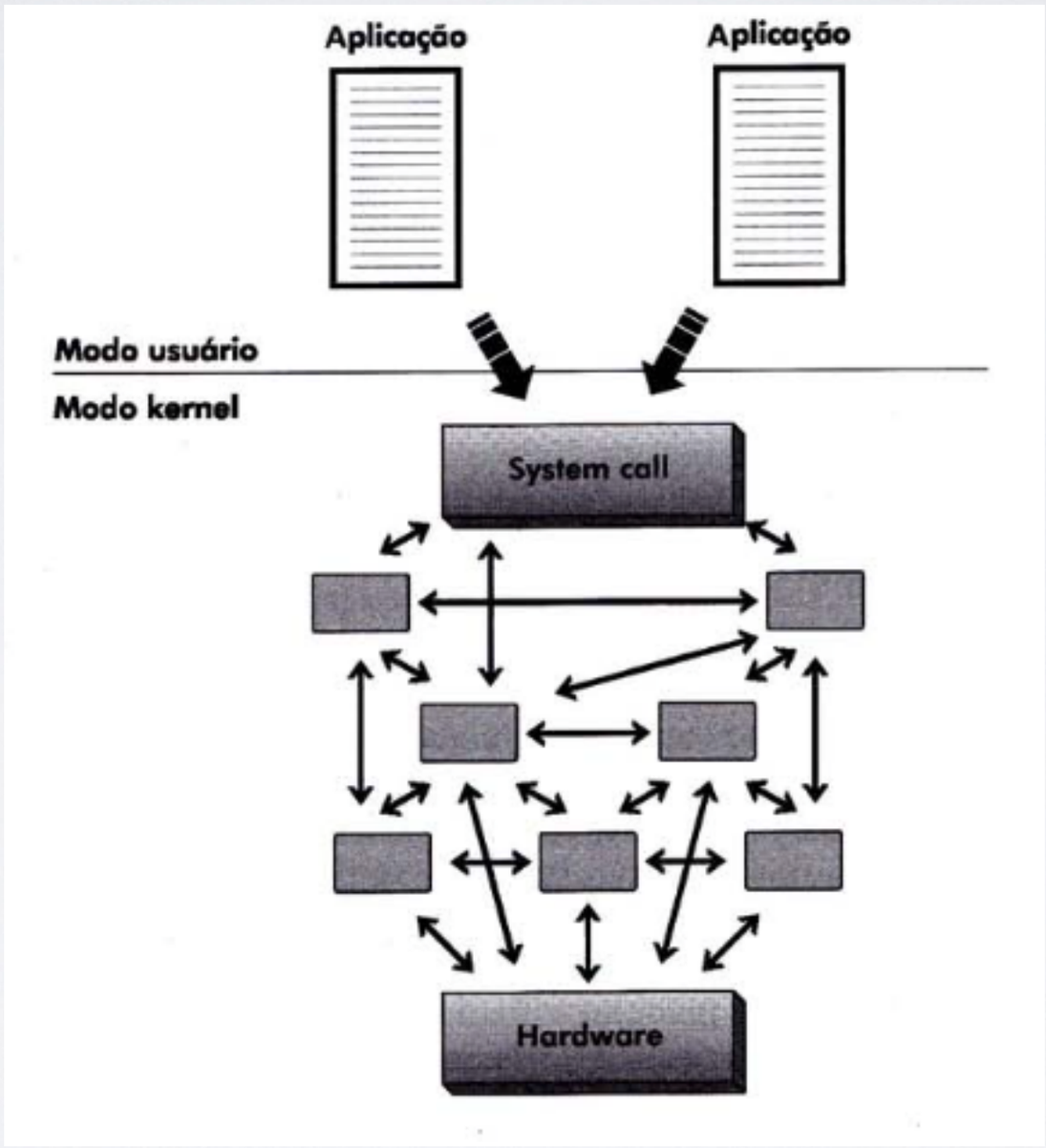
SYSTEM CALLS



NÚCLEO MONOLÍTICO

- Nesta abordagem o SO inteiro é executado como um único programa no modo núcleo;
- Pode ser comparada com uma aplicação formada por vários procedimentos que são compilados separadamente e de pois linkados, formando um grande e único programa executável. Estes "procedimentos" são chamados de módulos.
- Grande desempenho
- Uma falha pode paralisar todo o núcleo. O sistema pode parar por causa de um erro.
- Seu uso: Linux, BSD, Windows

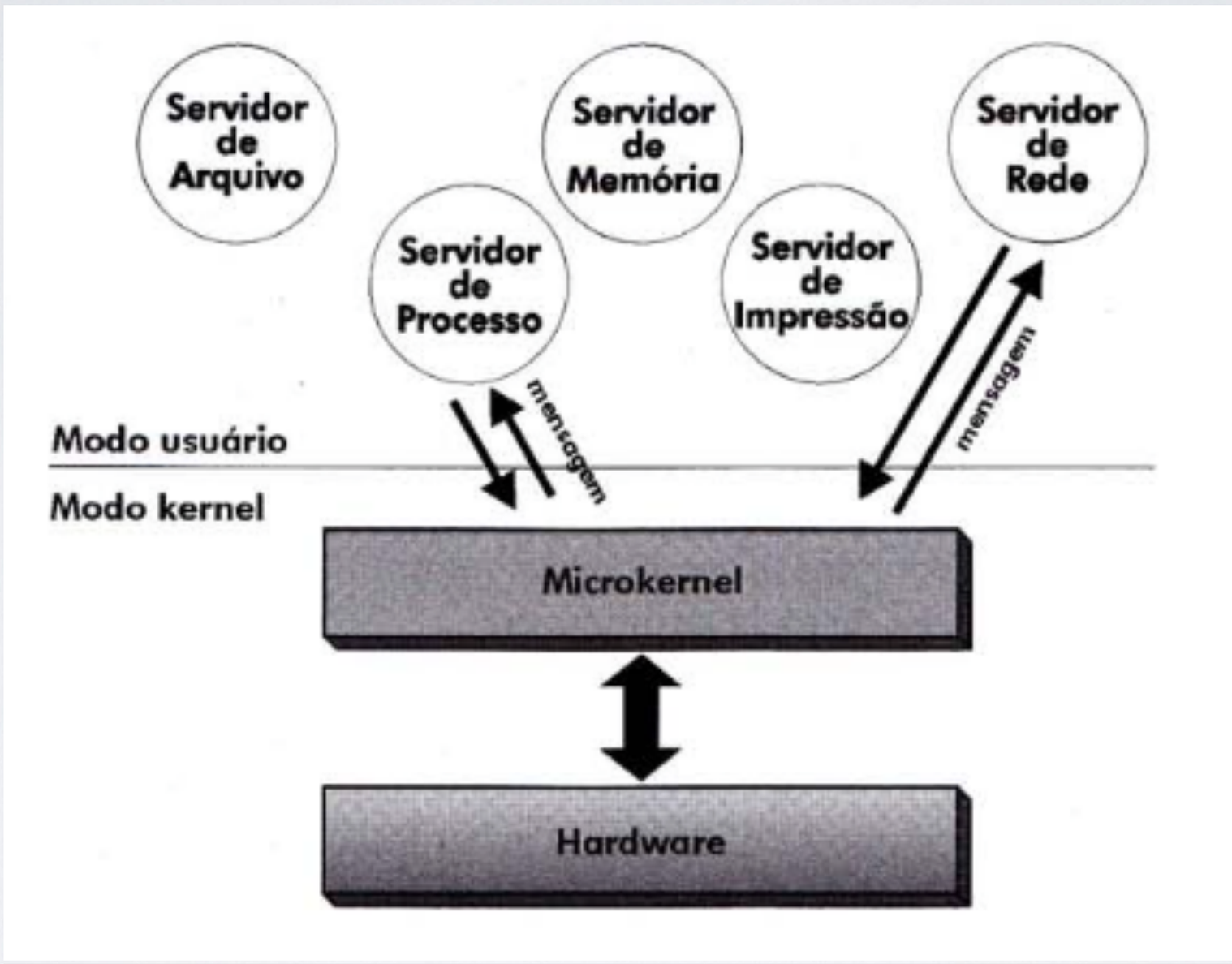
NÚCLEO MONOLÍTICO



MICROKERNEL

- Arquitetura de kernel que possui apenas um núcleo que provê recursos mínimos necessários ao ambiente;
- Outras funcionalidades são oferecidas através de programas chamados servidores, que se localizam na user-space;
- Proporciona maior segurança;
- Facilita a manutenção;
- Exemplos: Integrity, K42, PikeOS, QNX, Symbiam e MINIX3.

MICROKERNEL



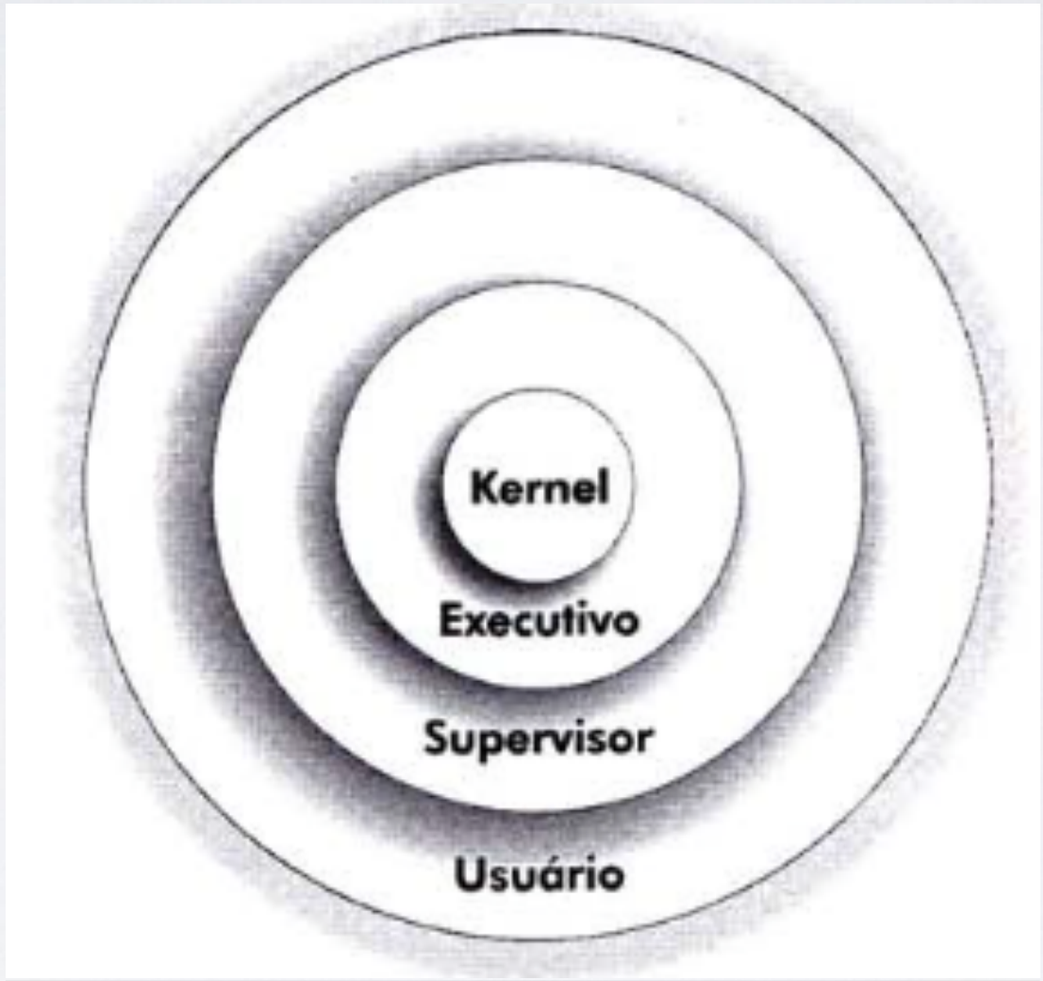
SISTEMAS EM CAMADAS

- Um sistema em camadas torna o papel do S.O. e do núcleo dividido entre diversos níveis, permitindo a cada nível se aproximar ainda mais do hardware;
- Existe alguma perda de desempenho em caso de camadas serem postas indiscriminadamente;
- Exemplos: A camada de HAL – Hardware Abstraction Layer (Windows NT em diante). IBM OS/2 e o MULTICS.
- O sistemas operacionais atuais em sua maioria utiliza essa arquitetura com duas camadas: nível de usuário e nível de núcleo, conforme visto anteriormente.

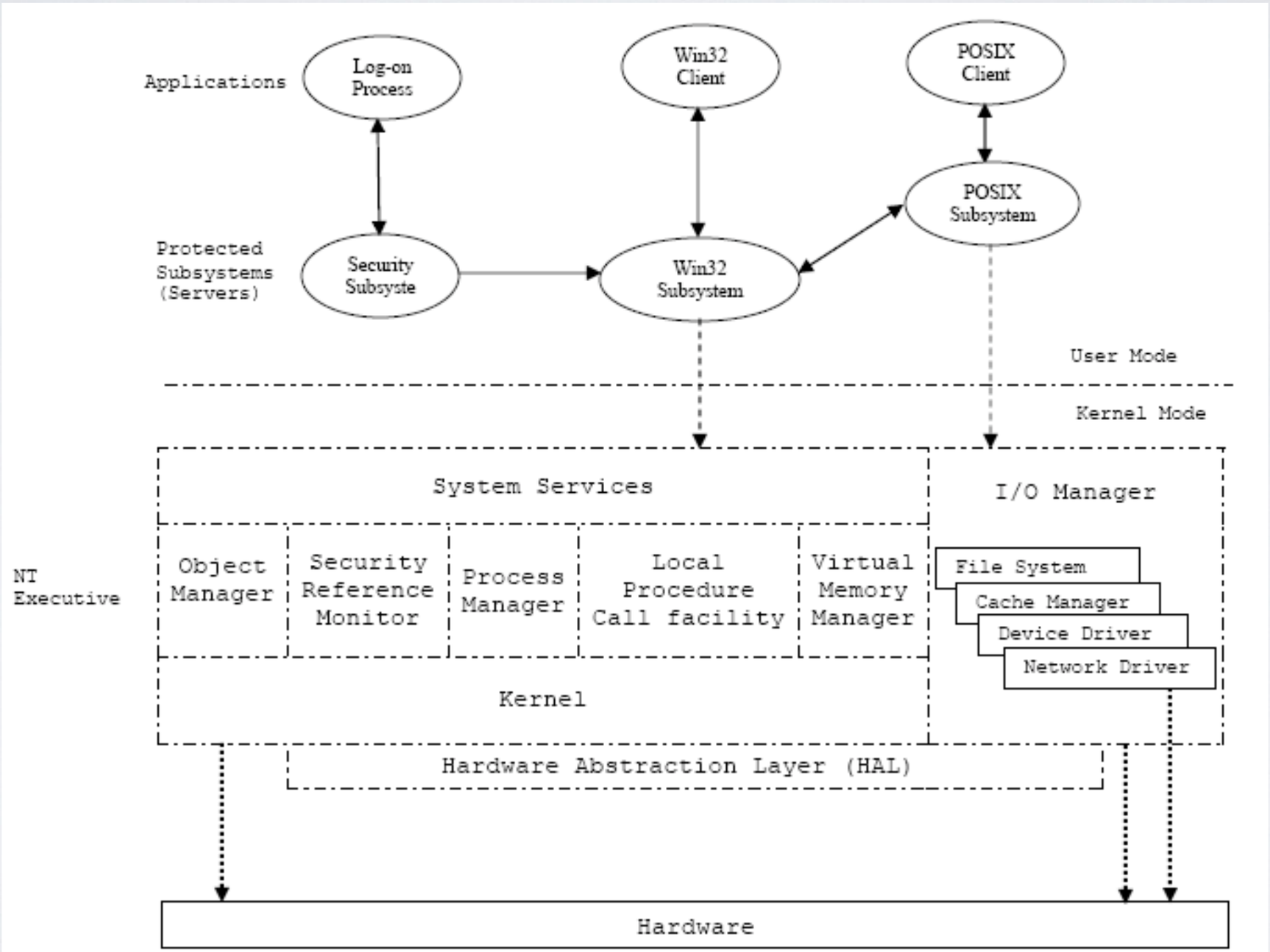
SISTEMAS EM CAMADAS

- HAL: Hardware Abstraction Layer, camada de comunicação com o hardware;
- Kernel: sistema primários como escalonador;
- Subsistemas;
- Serviços de sistema, incluindo utilitários.

SISTEMA EM CAMADAS



SISTEMA EM CAMADAS



ENFIM!

- Qual a arquitetura atual?

- Hoje temos sistemas baseados no conceito de micro-kernel e sistema em camadas;

- Foi visto que desenvolver todos a gerências do S.O. em modo usuário (sem acesso a instruções privilegiadas) trazia grande dificuldade e baixo desempenho;

- Hoje temos algumas gerências implementadas a nível de kernel (memória, escalonamento,, e/s) e serviços mais simples (servidor de impressão, interface gráfica, etc) a nível de usuário;

REFERÊNCIAS

MACHADO, Francis B.; MAIA, Luiz Paulo. Arquitetura de Sistemas Operacionais. 3a ed. Rio de Janeiro : LTC, 2002.

DÚVIDAS?