Sistemas Operacionais

Machado/Maia

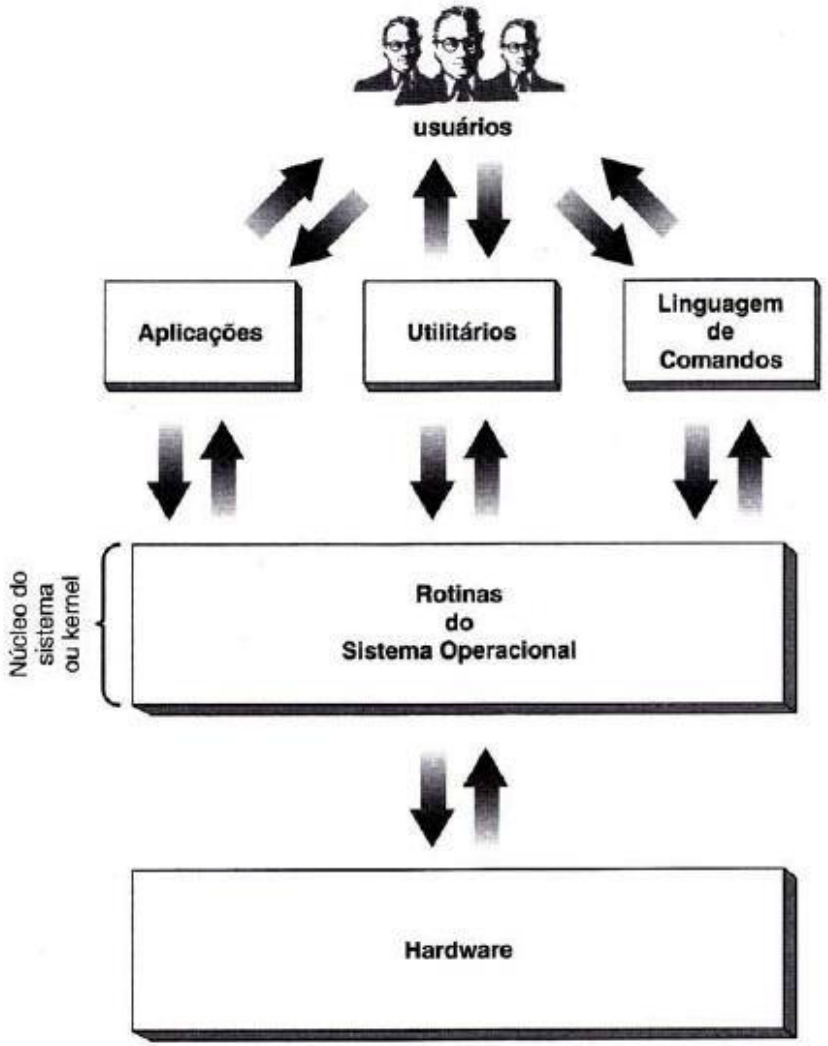
Prof. Dr. Ricardo Ramos

Capítulo 04
Estrutura dos SOs

4.1 Introdução

- O *kernel* (núcleo do sistema operacional) é um conjunto de rotinas que oferecem serviços aos usuários e às suas aplicações.

4.1 Introdução



4.2 Funções do Núcleo

As principais funções do *kernel* são:

- gerência de processos e threads;
- gerência de memória;
- gerência do sistema de arquivos;
- gerência de dispositivos de E/S;
- e outras.

4.3 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 a seus serviços.

Modos de acesso (mecanismo de segurança presente no hardware dos processadores)

4.3 Modo de Acesso

Modo usuário:

Uma aplicação só pode executar instruções não-privilegiadas (não oferecem riscos ao sistema), tendo acesso a um número reduzido de instruções.

Modo kernel:

A aplicação pode ter acesso ao conjunto total de instruções do processador, inclusive as privilegiadas (podem ocasionar sérios problemas à integridade do sistema se usadas de maneira indiscriminada).

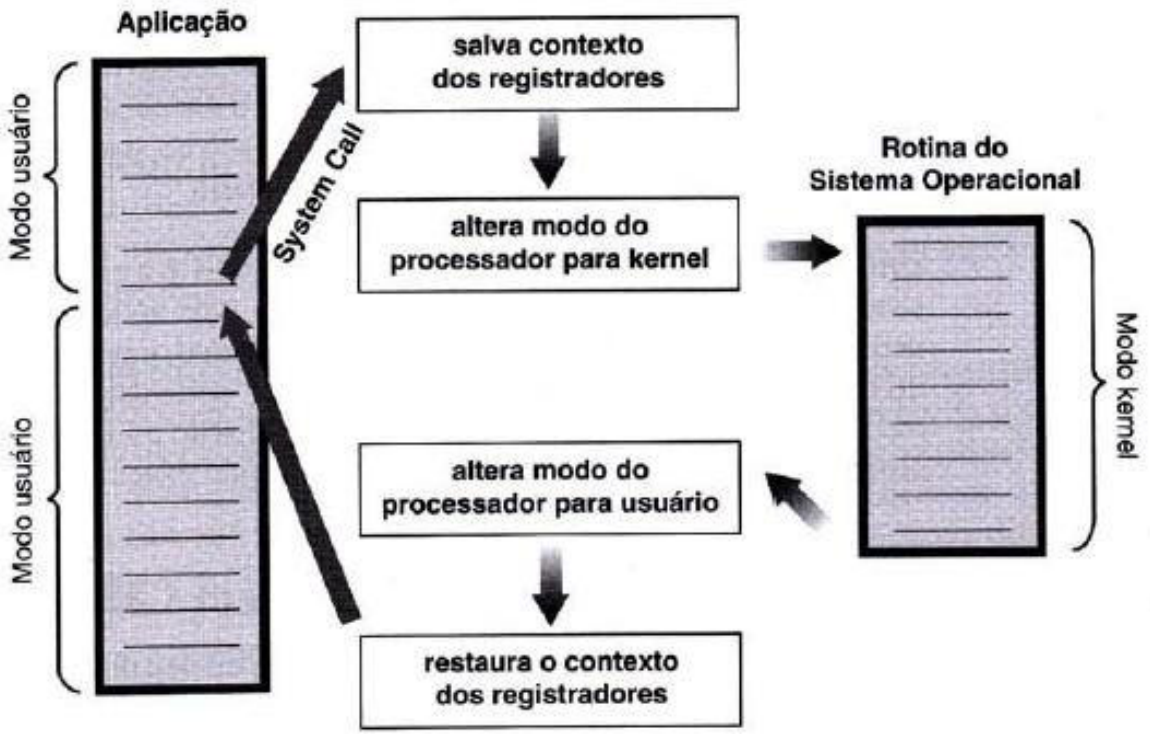
4.4 Rotinas do SO e *System Calls*

Todas as funções do núcleo são implementadas por rotinas do sistema que necessariamente possuem em seu código instruções privilegiadas.

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

4.4 Rotinas do SO e *System Calls*

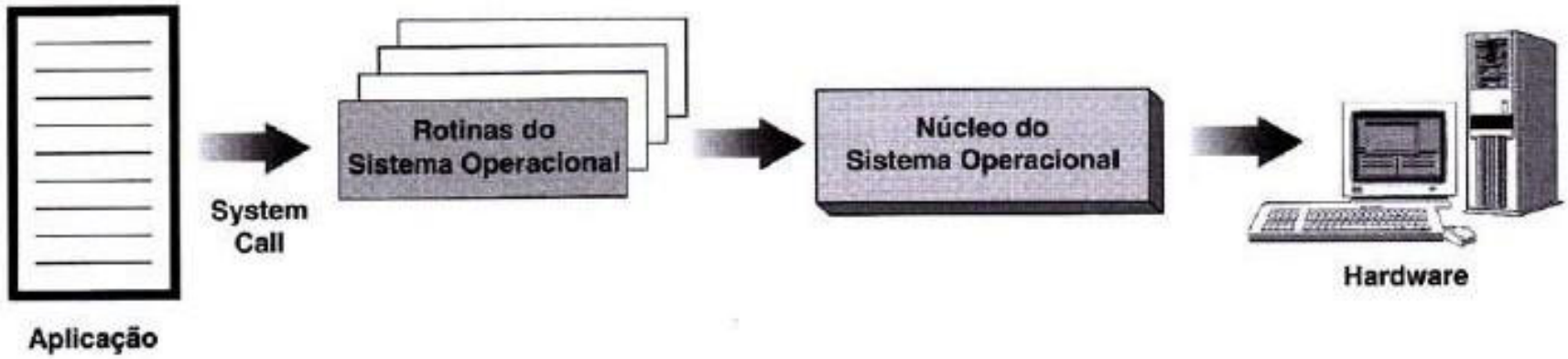
(porta de entrada para o *kernel*)



Ex: Acesso ao disco rígido (operação de E/S).

4.5 Chamada a Rotinas do SO

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



Semelhante a chamada de uma sub-rotina por um programa.

4.5 Chamada a Rotinas do SO

Cada SO possui seu próprio conjunto de rotinas, com nomes, parâmetros e formas de ativação específicos.

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

4.6 Linguagem de Comandos

(ou linguagem de controle)

Permite comunicação simples com o SO.

O usuário dispõe de uma interface direta com o SO.

Tabela 4.2 Exemplos de Comandos do MS Windows

Comando	Descrição
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

4.6 Linguagem de Comandos

(ou linguagem de controle)

Cada comando, depois de digitado pelo usuário, é interpretado pelo *shell* ou **interpretador de comando** (em geral, não faz parte do SO), que verifica a sintaxe do comando, faz chamadas a rotinas do sistema e apresenta um resultado ou uma mensagem informativa.

No Linux, o *shell* padrão é o bash.

4.6 Linguagem de Comandos

(ou linguagem de controle)