UNIP
UNIVERSIDADE PAULISTA

Sistemas Operacionais

Estrutura SOs – 5º Período

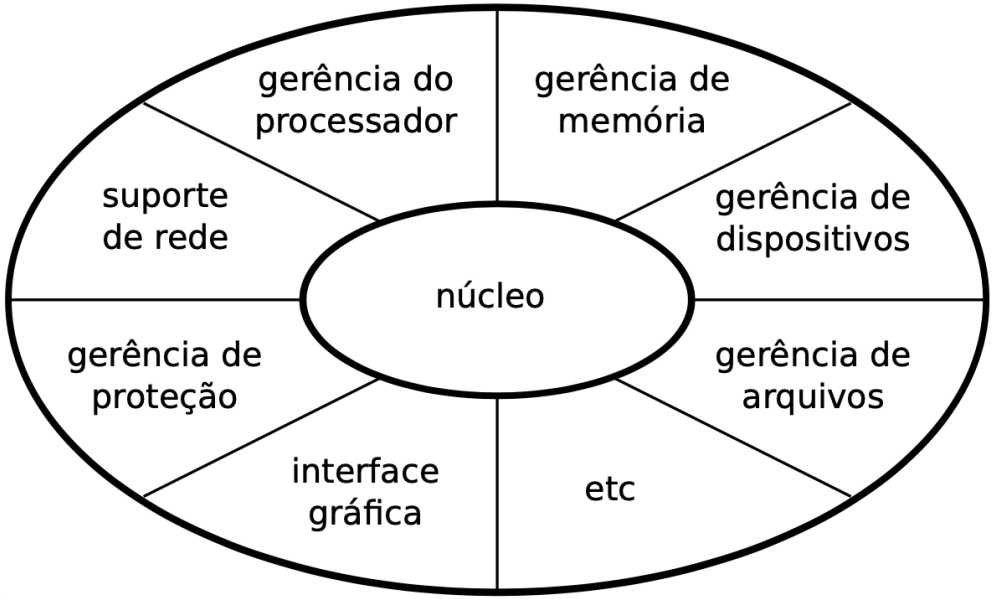
UNIP
UNIVERSIDADE PAULISTA

Roteiro

- Funcionalidades
- Sistemas Monolíticos
- Sistemas em Camadas
- Sistemas Micro-núcleo
- Máquinas Virtuais
- Exonúcleo

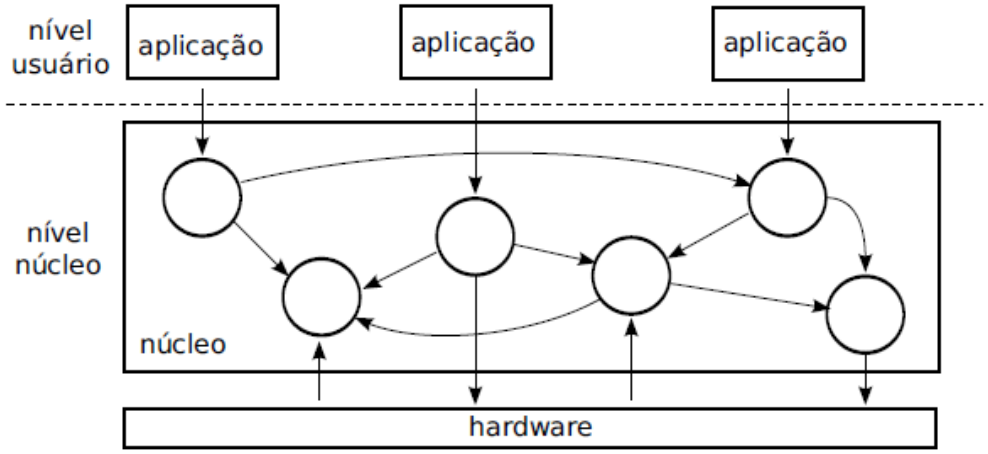
Funcionalidades

- Para cumprir seus objetivos de abstração e gerência, o sistema operacional deve atuar em várias frentes. Cada um dos recursos do sistema possui suas particularidades, o que impõe exigências específicas para gerenciar e abstrair os mesmos. Sob esta perspectiva, as principais funcionalidades implementadas por um sistema operacional típico são:



Sistemas Monolítico

- Nesta arquitetura todos os componentes do SO são executados como um **único programa no modo núcleo** e se inter-relacionam conforme suas necessidades, sem restrições de acesso entre si, pois o código no nível núcleo tem acesso pleno a todos os recursos e áreas de memória.
- A grande vantagem dessa arquitetura é seu **desempenho**: qualquer componente do núcleo pode acessar os demais componentes, toda a memória ou mesmo dispositivos periféricos diretamente, pois não há barreiras impedindo esse acesso. A interação direta entre componentes também leva a sistemas mais compactos.



Sistemas Monolítico

- Todavia, a arquitetura monolítica pode pagar um preço elevado por seu desempenho: nada robusto e nem um pouco fácil de desenvolver. Caso um componente do núcleo perca o controle devido a algum erro, esse problema pode se **alastrar** rapidamente por todo o núcleo, levando o sistema ao **colapso** (travamento, reinicialização ou funcionamento errático). Além disso, a manutenção e evolução do núcleo se tornam mais complexas, porque as dependências e pontos de interação entre os componentes podem não ser evidentes: pequenas alterações na estrutura de dados de um componente podem ter um impacto inesperado em outros componentes, caso estes acessem sua estrutura diretamente.
- A arquitetura monolítica foi a primeira forma de organizar os sistemas operacionais; sistemas UNIX antigos e o MS-DOS seguiam esse modelo.

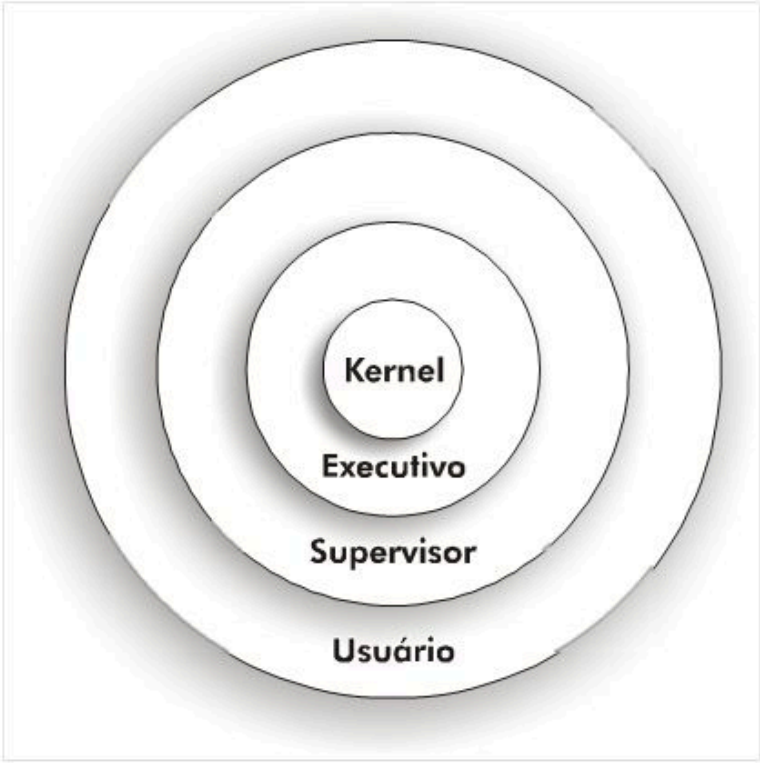
Sistemas em Camadas

- Divide o sistema operacional em sistemas sobrepostos. Cada módulo oferece um conjunto de funções que podem ser chamadas por outros módulos.
  - O sistema MULTICS VMS as camadas inferiores são as mais privilegiadas.
- A vantagem da estruturação em camadas é **isolar o sistema operacional**, facilitando sua alteração e depuração, além de criar uma hierarquia de níveis de modos, protegendo as camadas mais internas.

Sistemas em Camadas

Camada	Função
5	O operador
4	Programas de usuário
3	Gerenciamento de entrada/saída
2	Comunicação operador–processo
1	Memória e gerenciamento de tambor
0	Alocação do processador e multiprogramação

Sistema THE



MULTICS VMS

Sistema em Camadas

- O sistema THE era um sistema de lote simples para um computador holandês, o Electrologica X8.
- Na camada 5 – estava localizado o processo operador do sistema.
- Na camada 4 – localizavam-se os programas de usuários. Eles não tinham de se preocupar com o gerenciamento de processo, memoria, console ou E/S.
- Camada 3 – gerenciava dispositivos de entrada e saída.
- Camada 2 – fazia a comunicação entre o console do operador e cada processo.
- Camada 1 – fazia o gerenciamento da memória. Ela alocava espaço para os processos da memória principal e em um tambor (Antigo meio magnético de armazenamento de dados) utilizado para armazenar partes do processo (páginas) para os quais não havia lugar na memória principal.
- Camada 0 – lidava com alocação do processador , alternando entre processos quando ocorriam interrupções ou quando os temporizadores expiravam. Em outras palavras, a camada 0 (zero) proporcionava a multiprogramação básica da CPU.

Sistemas em Camadas

- O empilhamento de várias camadas de software faz com que cada pedido de uma aplicação **demore mais tempo** para chegar até o dispositivo periférico ou recurso a ser acessado, prejudicando o desempenho do sistema.
- Não é óbvio dividir as funcionalidades de um núcleo de sistema operacional em **camadas horizontais** de abstração crescente, pois essas funcionalidades são interdependentes, embora tratem muitas vezes de recursos distintos.

Sistemas em Camadas

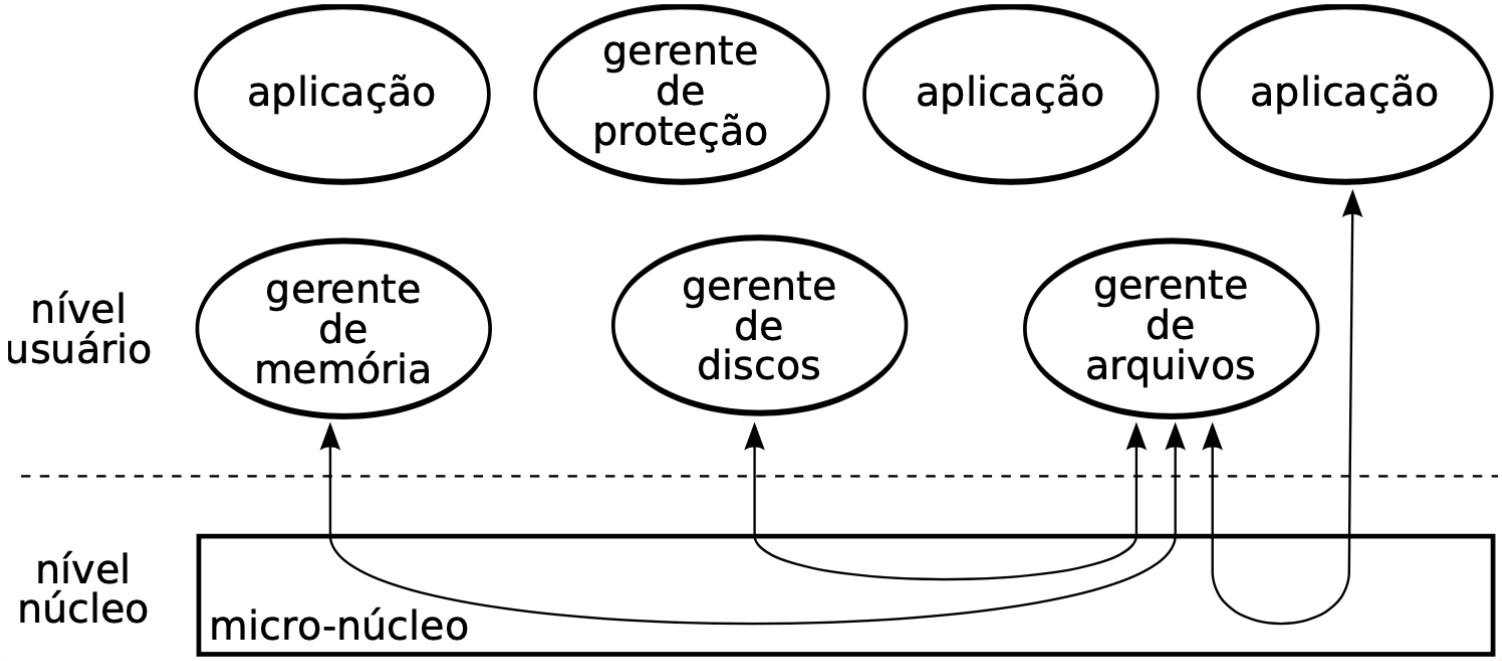
- Em decorrência desses inconvenientes, a estruturação em camadas é apenas parcialmente adotada hoje em dia. Muitos sistemas implementam uma camada inferior de **abstração do hardware** para interagir com os dispositivos (a camada *HAL – Hardware Abstraction Layer*, implementada no Windows NT e seus sucessores), e também organizam em camadas alguns subsistemas como a **gerência de arquivos** e o **suporte de rede**.

Sistemas Micro-Núcleo (μ-*kernel*)

- Uma tendência dos sistemas operacionais é tornar o núcleo menor e mais simples possível, no entanto para implementar esta ideia o sistema é dividido em **processos**.
- Desta forma, sempre que uma aplicação deseja algum serviço ela solicita ao processo responsável, assim, a aplicação que solicita um serviço é chamada de **cliente** e o processo que responde a solicitação é chamado de **servidor**.
- A utilização deste modelo permite que os servidores executem em **modo usuário** e apenas o núcleo do sistema, responsável pela comunicação entre clientes e servidores, executa no **modo kernel**.

Sistemas Micro-Núcleo

- Todos os aspectos de alto nível, como políticas de uso do **processador** e da **memória**, o **sistema de arquivos** e o **controle de acesso** aos recursos são implementados fora do núcleo, em processos que se comunicam usando as **primitivas do núcleo**.

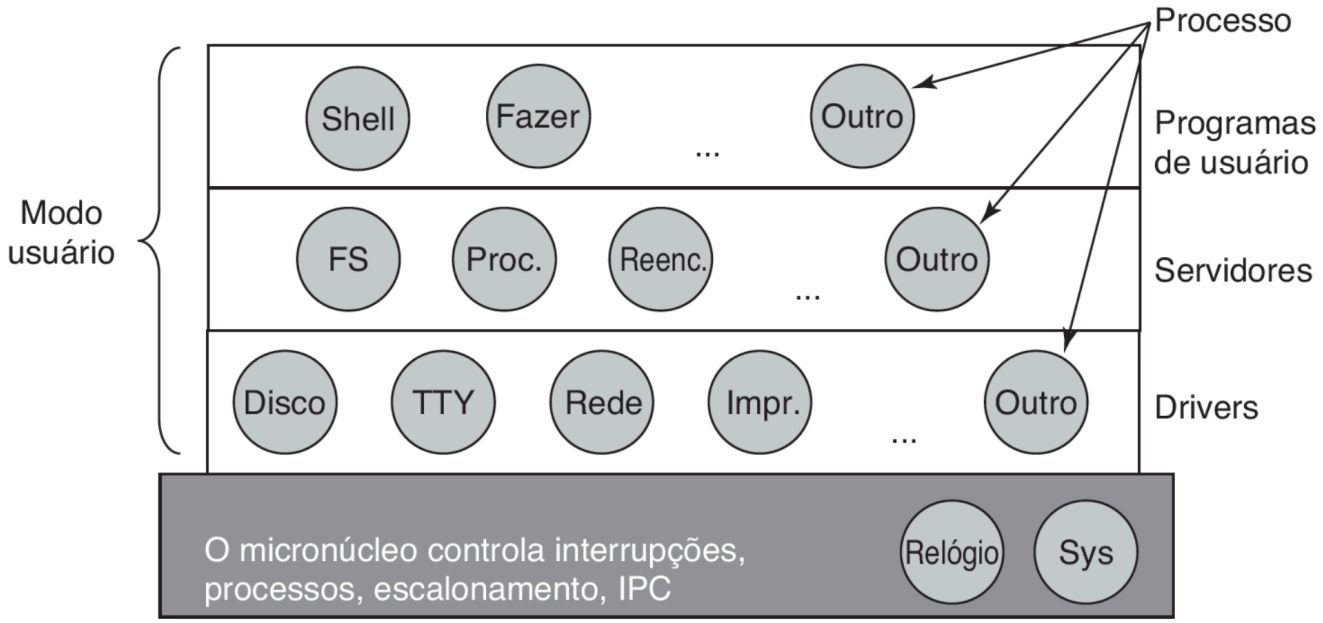


Sistemas Micro-Núcleo

- As principais vantagens dos sistemas micro-núcleo são sua **robustez e flexibilidade**: caso um sub-sistema tenha problemas, os mecanismos de proteção de memória e níveis de privilégio irão confiná-lo, impedindo que a instabilidade se alastre ao restante do sistema. Além disso, é possível customizar o sistema operacional, iniciando somente os componentes necessários ou escolhendo os componentes mais adequados às aplicações que serão executadas.
- O sistema operacional passa a ser de mais fácil manutenção.
- Não importa se o serviço esta sendo processado em um único processador, com múltiplos processadores (fortemente acoplado) ou em sistema distribuído (fracamente acoplado).

Micro-Núcleo

- Micronúcleos são comuns em aplicações de tempo real, industriais, avionica e militares, que são cruciais e tem requisitos de confiabilidade muito altos.
- Ex: Integrity, K42, PikeOS, QNX, Symbiam e MINIX3.

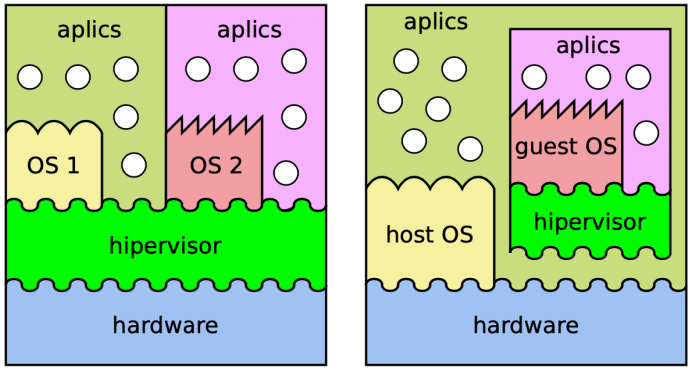


Máquina Virtual

- Maquina Virtual é uma camada de software que emula as características de interfaces distintas, de forma que um programa desenvolvido para uma plataforma *A* possa executar sobre uma plataforma distinta *B*.
- São cópias exatas do hardware, inclusive com modos núcleo/usuário, E/S, interrupções e tudo o que uma máquina real tem.
- Cada VM pode executar qualquer SO capaz de ser executado diretamente sobre o hardware.
- Diferentes VMs podem executar diferentes SOs.

Máquina Virtual

- **Hipervisores nativos** (ou de tipo I): o hipervisor executa diretamente sobre o hardware da máquina real, sem um sistema operacional subjacente. A função do hipervisor é multiplexar os recursos de hardware (memória, discos, interfaces de rede, etc) de forma que cada máquina virtual veja um conjunto de recursos próprio e independente. Alguns exemplos de sistemas que empregam esta abordagem são o IBM OS/370, o VMWare ESX Server e o ambiente Xen.
- **Hipervisores convidados** (ou de tipo II): o hipervisor executa como um processo normal sobre um sistema operacional hospedeiro. O hipervisor usa os recursos oferecidos pelo sistema operacional real para oferecer recursos virtuais ao sistema operacional convidado que executa sobre ele. Exemplos de sistemas que adotam esta estrutura incluem o VMWare Workstation, o QEmu e o VirtualBox.



hipervisor nativo hipervisor convidado

Exonúcleo

- **Exonúcleo** é um paradigma de implementação em que existe em um núcleo simples que faz gerência de recursos do sistema e um conjunto de bibliotecas que implementam a abstração de um sistema operacional. O aplicativo pode utilizar **diretamente** os recursos dos dispositivos do sistema ou pode utilizar bibliotecas de software e são capazes de compartilhar o mesmo ***hardware*** simultaneamente, similarmente a uma **máquina virtual**. A diferença é que em um sistema exonúcleo o *hardware* é alocado previamente, não podendo ocorrer o acesso indevido de outro sistema.
- Em vez de clonar a máquina real, como nas VMs, outra **estratégia é dividi-la** ou dar a cada usuário um subconjunto de recursos. Assim uma VM pode ter os blocos de 0 a 1023 do disco e outra de 1024 a 2047 e assim por diante.
- Na camada inferior há o programa exonúcleo, responsável por alocar recursos às VMs e verificar as tentativas de uso para assegurar que uma máquina não esteja utilizando os **recursos de outra.**
- Cada VM pode utilizar seu próprio SO mas **somente** com os recursos que pediu e que foram alocados.
- Este tem vantagem de separar, com **menor custo**, a multiprogramação (no exonúcleo) do código do SO do usuário, visto que o exonúcleo mantém as VMs umas fora do alcance das outras. Mais confiabilidade e segurança.

Dúvidas?

Prof. Mateus Leandro Paulino
Também conhecido por Matox
mateus.paulino@docente.unip.br
classroom code: jmwi26g

Assim também Cristo foi oferecido uma só vez em sacrifício, para tirar os pecados de muitas pessoas. Depois ele aparecerá pela segunda vez, não para tirar pecados, mas para salvar as pessoas que estão esperando por ele. Hebreus 9.28