Sistemas Operacionais I

Arquitetura dos Sistemas Operacionais

Prof. Carlos Eduardo de B. Paes
Departamento de Ciência da Computação
PUC-SP

Estrutura dos SOs

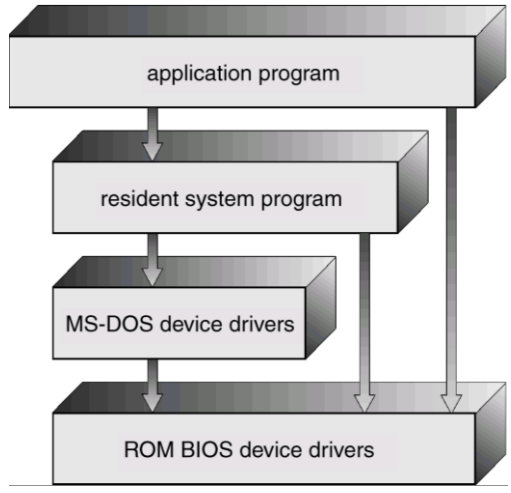
- Como estruturar um sistema operacional ?
- Vamos dar uma olhada em algumas estruturas possíveis que vão nos dar uma boa visão de alguns projetos de SO desenvolvidos na prática

Estrutura Simples

- Não possuem uma estrutura bem definida
- Sistemas que começaram pequenos e cresceram ao logo do tempo
- Ex: MS-DOS

3

Estrutura Simples



4

Sistemas Monolíticos

- Considerada uma "grande bagunça"
- Não existe estruturação nenhuma
- SO → basicamente um conjunto de procedimentos com interfaces bem definidas em termos de parâmetros
- Código objeto do SO → todos os procedimentos devem ser compilados
- Em seguida, todos os módulos são ligados em uma única estrutura (Sistema Operacional)

5

Sistemas Monolíticos

- Este tipo de sistema pode ser estruturado no mínimo da seguinte forma:
  - Implementação de system calls
  - Chamadas TRAP
  - Dois modos de operação:
    - Modo usuário
    - Modo kernel (ou supervisor)

6

Sistemas Monolíticos

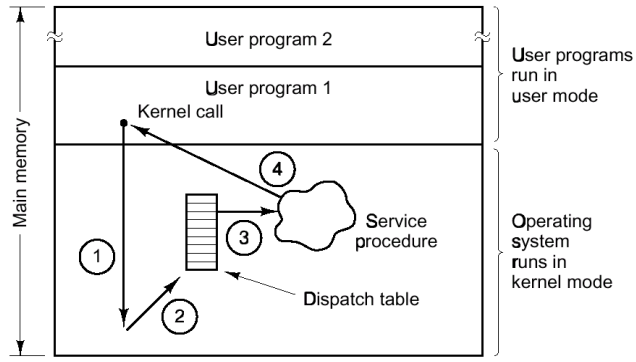


Figure 1-16. How a system call can be made: (1) User program traps to the kernel. (2) Operating system determines service number required. (3) Operating system calls service procedure. (4) Control is returned to user program.

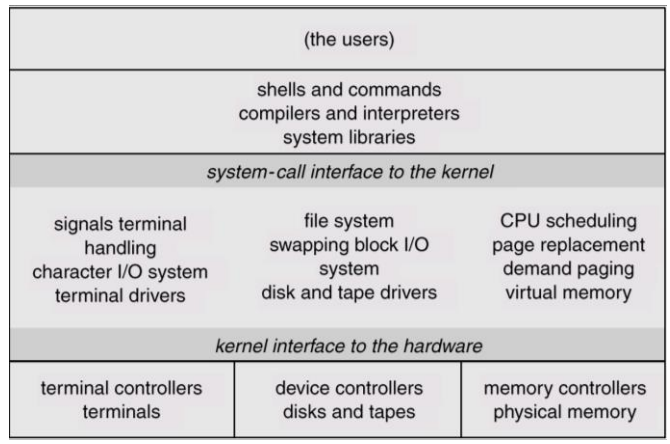
7

Sistemas Monolíticos

- UNIX: limitado pelas funcionalidades dos hardware, o Unix original tinha uma estruturação limitada
- O UNIX consistia em duas parte:
  - Programas de sistema
  - Kernel

8

Sistemas Monolíticos



9

Sistemas Monolíticos

- Com base no modelo apresentado podemos sugerir a seguinte estrutura básica:
  - Um programa principal que chama o procedimento de serviço
  - Um conjunto de procedimentos de serviço que executam cada uma das chamadas do sistema
  - Um conjunto de procedimento utilitários que auxiliam na execução do procedimentos de serviço

10

Sistemas Monolíticos

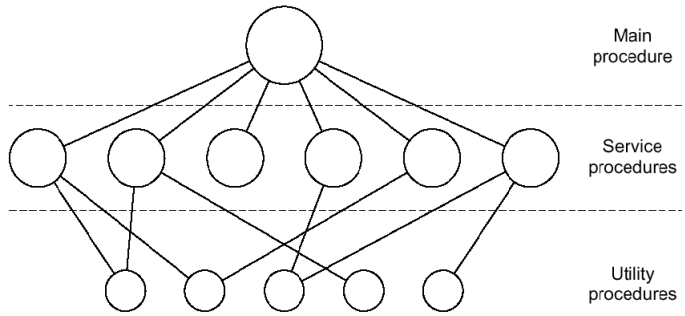


Figure 1-17. A simple structuring model for a monolithic system.

11

Sistemas em Camada

- Generalização do modelo da figura anterior
- O SO é organizado em hierarquia de níveis, cada um construído sobre o nível imediatamente abaixo
- Sistema Operacional THE (*Technishe Hogeschool Eindhoven*)
  - E. W. Dijkstra, 1968 – Holanda
  - Sistema batch muito simples
  - Electrologica X8

12

Sistemas em Camada
Camadas do Sistema *THE*

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

Figure 1-18. Structure of the THE operating system.

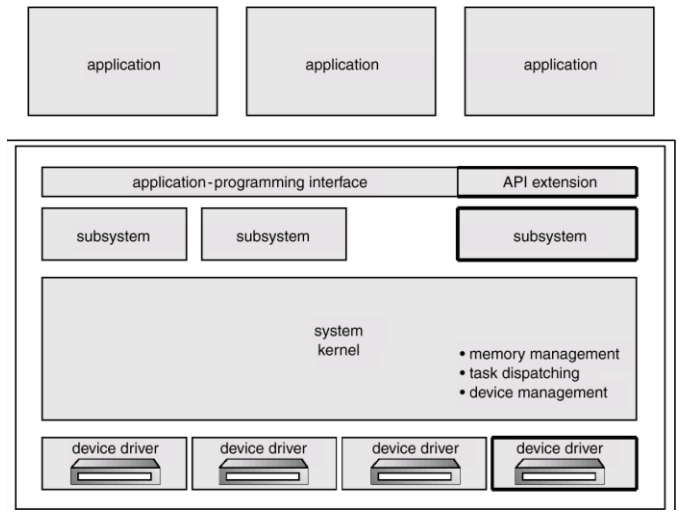
13

Sistemas em Camada

- Vantagem: modularidade
- Desvantagem: tende a ser menos eficiente que as outras abordagens
  - Quantidade de interações entre as camadas
- Ex: OS/2 da IBM
  - Descendente do MS-DOS
  - Usa a abordagem em camadas
  - Minimiza o problema da eficiência definindo menos camadas com mais funcionalidades

14

Sistemas em Camada
Camadas do OS/2



15

Sistemas em Camada

- Outro exemplo: Windows NT
  - Primeira versão → tinha uma organização fortemente orientada a camadas
    - Problema: péssimo desempenho
  - Windows NT 4.0: resolveu o problema parcialmente movendo camadas do espaço de endereçamento do usuário para o kernel e provendo maior integração entre elas.

16

Sistemas em Camada

- Sistema Operacional MULTICS → implementava uma generalização do conceito de níveis:
  - Usava anéis concêntricos (camadas)
  - Mais interno, mais privilegiado
  - Chamadas de sistemas eram geradas para as camadas mais internas (TRAP)

17

Microkernel

- Problema: Unix → o kernel começou a crescer e tornou-se grande e difícil de gerenciar.
- Princípio básico → este método estrutura o SO removendo todos os componentes não-essenciais do kernel e implementando-os como programa de sistema no nível de usuários.
- Resultado → Kernel menor

18

Microkernel

- Problemas: definir quais serviços devem permanecer no kernel e quais devem ser implementados no espaço de usuário.
- Sistema Operacional **Mach**
  - Surgiu na década de 80
  - Universidade de Carnegie Mellon
  - Usa abordagem de mircrokernel

19

Microkernel
Modelo Cliente-Servidor

- Solicitação de serviços são realizados no modelo cliente-servidor:
  - Processo cliente
  - Proesso servidor (serviços)
  - Kernel → implementa a comunicação entre clientes e servidores

20

Microkernel
Modelo Cliente-Servidor

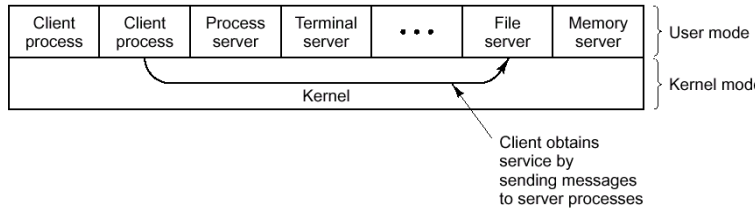


Figure 1-20. The client-server model.

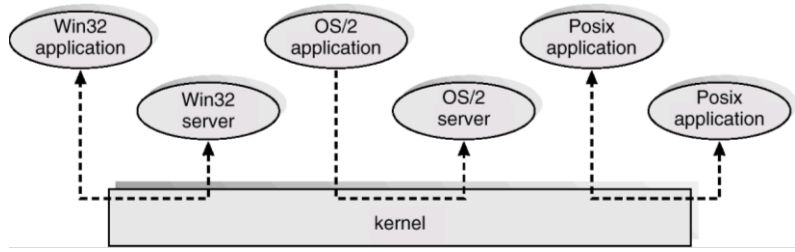
21

Microkernel
Modelo Cliente-Servidor

- Exemplos de Sistemas Operacionais:
  - Unix da Digital (é implementado com um kernel Mach)
  - Apple Mac OS X Server: se baseia no kernel do Mach
  - Windows NT: usa uma estrutura híbrida
    - Modelo cliente e servidor para atender diferentes tipos de aplicação (Win32, OS/2 e Posix)

22

Windows NT
Modelo Cliente-Servidor



23

Modelo Cliente-Servidor

- Principal vantagem deste modelo → pode ser facilmente adaptado ao sistemas distribuídos
  - Cliente-servidor: mensagens podem ser transmitidas em uma rede

24

Modelo Cliente-Servidor

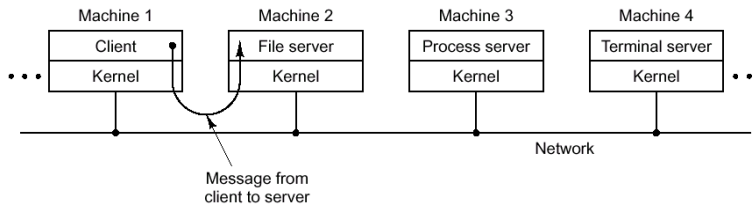


Figure 1-21. The client-server model in a distributed system.

25

Modelo Cliente-Servidor

- Problema: modelo não muito realista
  - Algumas funções do SO, tal como carga de comandos em registradores de dispositivos de I/O, são difíceis de ser implementado no espaço de endereçamento do usuário
  - Solução: implementar processos servidores críticos (*drivers* de dispositivos de I/O) rodando no modo *kernel*

26

Máquina Virtuais

- OS/360 → sistema batch
- Usuários → queriam um sistema de time-sharing
- IBM → desenvolveu um sistema TSS/360
- TSS/360 era um sistema muito grande e lento
- Centro Científico da IBM (Cambridge, Massachusetts) → desenvolveu um novo sistema chamado VM/370 (CP/CMS)

27

Máquina Virtuais

- VM/370 era um sistema de time-sharing:
  - Multiprogramado
  - Máquina estendida com uma interface mais conveniente que o hardware
  - O sistema separava estas duas coisas
- Coração do sistema → monitor da máquina virtual
  - Roda acima do hardware
  - Implementa a multiprogramação

28

Máquina Virtuais

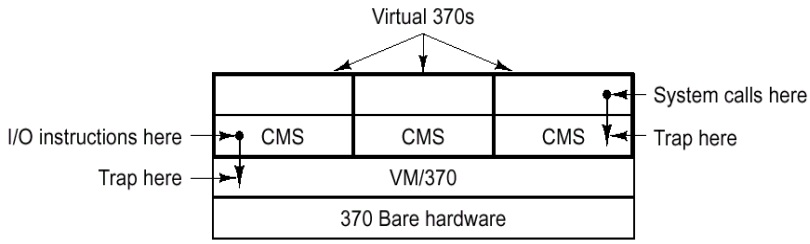
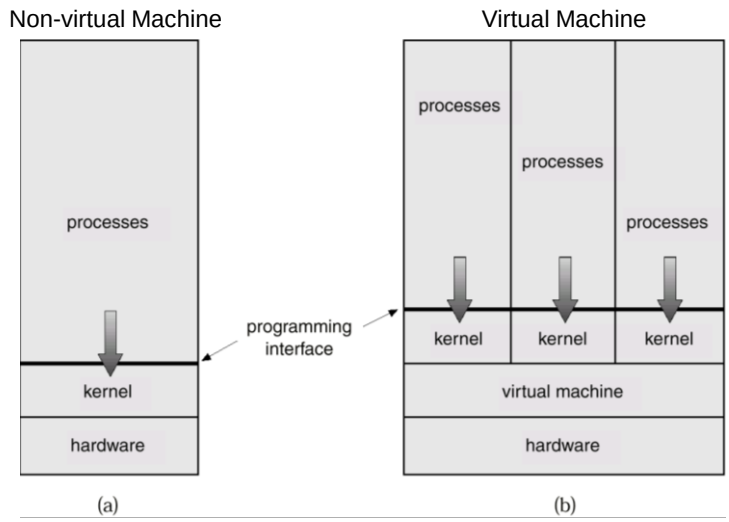


Figure 1-19. The structure of VM/370 with CMS.

29

Máquinas Virtuais



30

Máquina Virtuais

- A idéia de máquina virtual é intensamente utilizada hoje
  - Processadores Pentium → modo virtual 8086
  - Problema: questões de compatibilidade
  - Programas MS-DOS são executados em uma máquina virtual 8086 (emulação de um máquina 8086)

31

Máquina Virtuais

- Pergunta: Vocês conhecem outro sistema que usa amplamente este conceito?

Máquina Virtual JAVA

32

Objetivos do Projeto de SO

- **Objetivos p/ os usuários** → sistema operacional deve ser conveniente para usar, fácil de aprender, confiável, seguro e rápido
- **Objetivos do sistema** → sistema operacional deve ser fácil de projetar, implementar e manter, bem como flexível, confiável, livre de erros e eficiente

33

Mecanismos versus Políticas

- Mecanismos determina ***como*** fazer alguma coisa; Políticas determinam ***o que*** será feito.
- A separação entre política e mecanismo é um princípio muito importante, permite maximizar a flexibilidade se as decisões políticas precisam ser mudadas mais tarde.

34

Implementação do SO

- Tradicionalmente escrito em linguagem Assembly, SO atualmente podem ser escritos em linguagens de alto nível
- Códigos escritos em linguagens de alto nível
  - Acelera a implementação
  - É mais compacto
  - É mais fácil de entender e depurar
- Um SO é muito mais fácil de ser portado (movido para um outro hardware) se for escrito usando uma LP de alto nível

35

Geração do SO (SYSGEN)

- Um SOs são projetados para rodar em qualquer classe de máquina; o sistema deve ser configurado para cada computador específico
- Programa SYSGEN → obtém as informações da configuração específica do sistema de hardware
- ***Booting*** → inicializa o computador carregando o kernel do SO
- ***Bootstrap program*** → código armazenado em ROM que localiza o SO, carrega-o na memória, e inicializa sua execução

36