Infra-Estrutura de Software

Introdução

(cont.)

O que vimos

- Complexidade do computador moderno, do ponto de vista do hardware
- Necessidade de abstrações – software
- Sistema computacional em camadas
- SO como uma máquina estendida – abstrações
- SO como um gerenciador de recursos
  - Gerenciamento
  - Proteção
  - Compartilhamento

O que vimos

- Complexidade do computador moderno, do ponto de vista do hardware
- Necessidade de abstrações – software
- Sistema computacional em camadas
- SO como uma máquina estendida – abstrações
- SO como um gerenciador de recursos
  - Gerenciamento
  - Proteção
  - Compartilhamento

Um pouco de hardware…

- CPU: unidade de controle e execução
  - Registradores
  - Ciclo *fetch-decode-execute*
- Barramentos e dispositivos de E/S
- Memória [hierarquia]

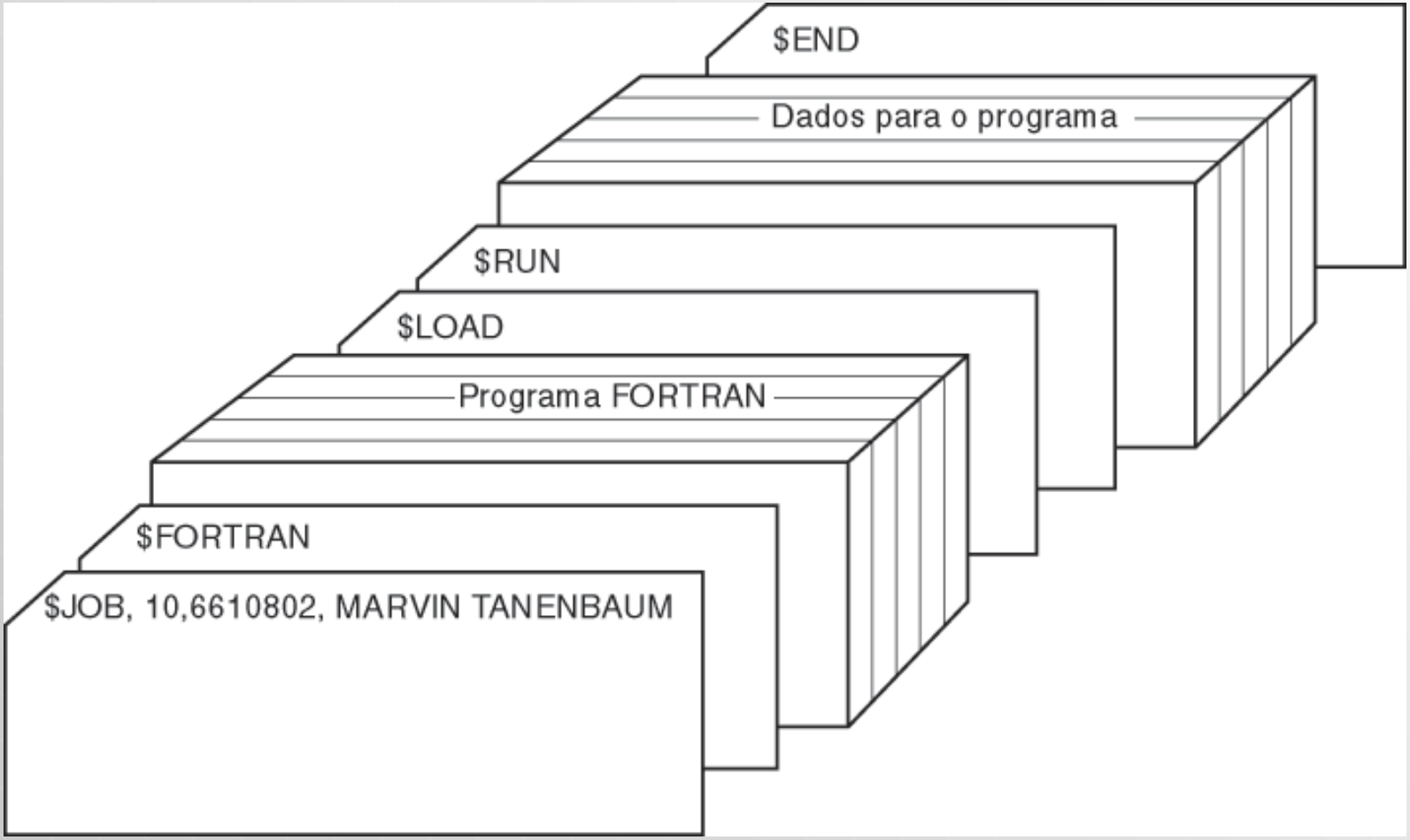
Software básico

- Linguagem de programação de alto nível ⇨ linguagem de montagem (Assembly) ⇨ linguagem de máquina
- Processo: um programa em execução
- Comunicação entre processos
- Sistemas Distribuídos

História dos Sistemas Operacionais

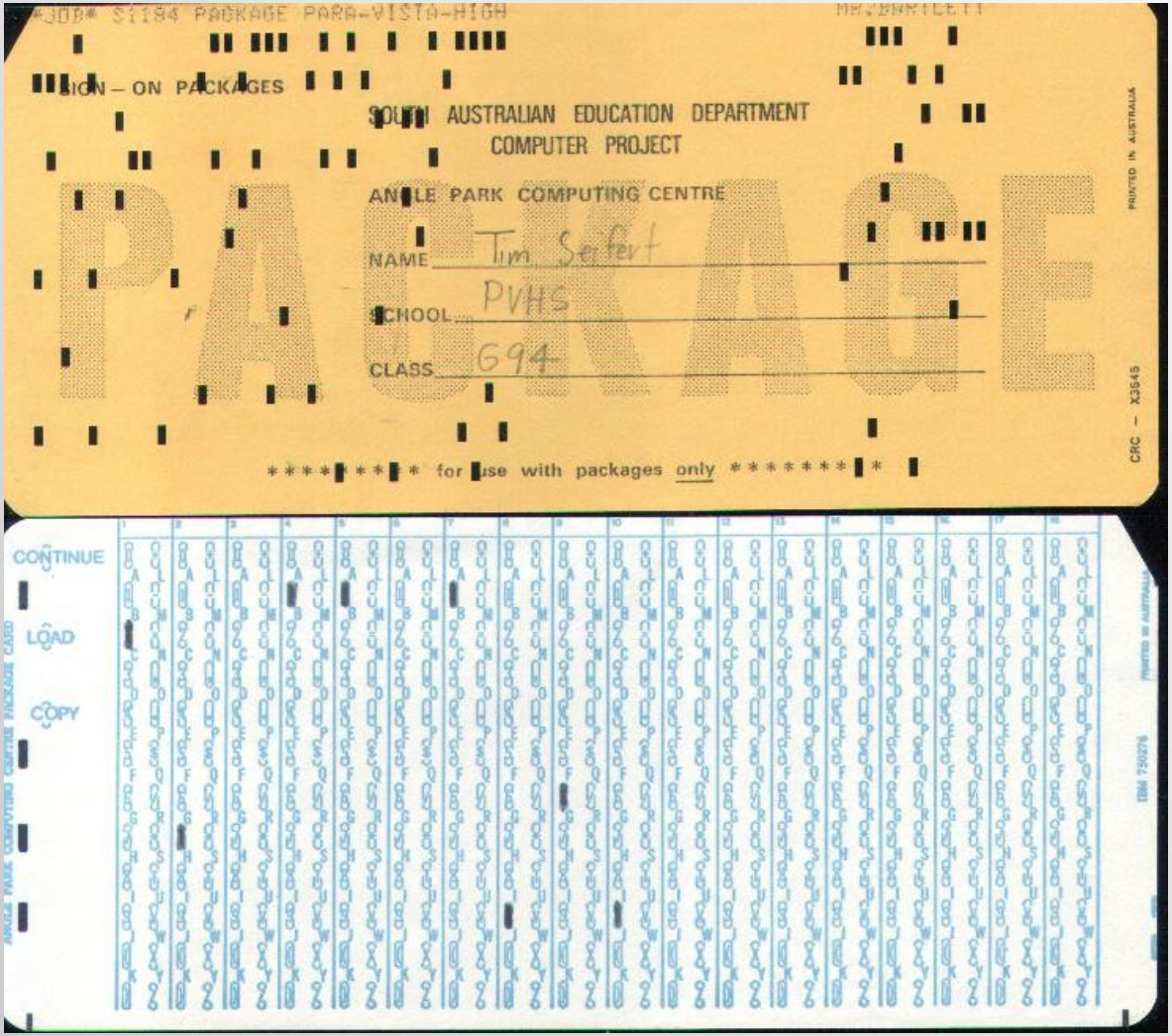
- Primeira geração: 1945 - 1955
  - Válvulas, painéis de programação
- Segunda geração: 1955 - 1965
  - transistores, sistemas em lote
- Terceira geração: 1965 – 1980
  - CIs (circuitos integrados) e multiprogramação
- Quarta geração: 1980 – presente
  - Computadores pessoais
- Hoje: onipresença – computação ubíqua

História dos Sistemas Operacionais

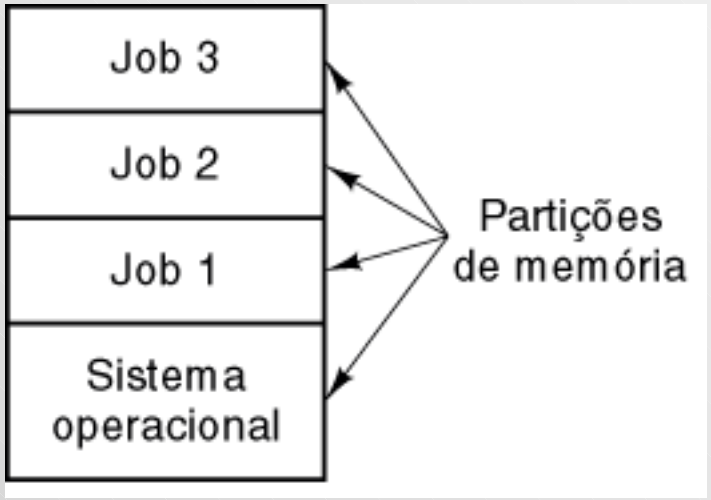


- Estrutura de um job típico (lote de cartões) – 2a. geração

Cartões perfurados
(*punched cards*)



História dos Sistemas Operacionais

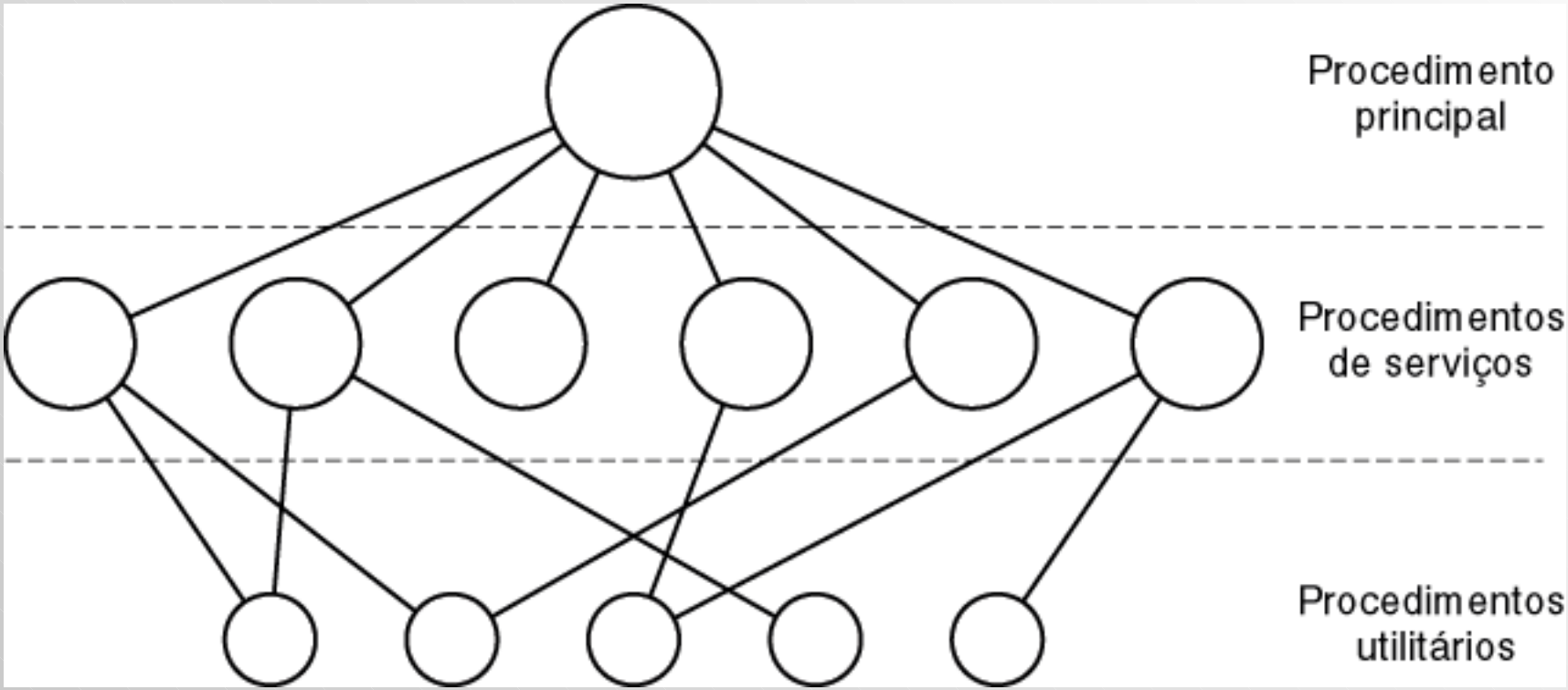


- Sistema de multiprogramação
  - Três jobs na memória – 3a. geração

Diversidade de Sistemas Operacionais

- Sistemas operacionais de computadores de grande porte (*mainframe*)
- Sistemas operacionais de servidores / redes
- Sistemas operacionais de multiprocessadores (paralelismo)
- Sistemas operacionais de computadores pessoais
- Sistemas operacionais de dispositivos portáteis/ móveis (ex. celulares)
- Sistemas operacionais de tempo-real
- Sistemas operacionais embarcados
- Sistemas operacionais de cartões inteligentes
- Sistemas operacionais de sensores

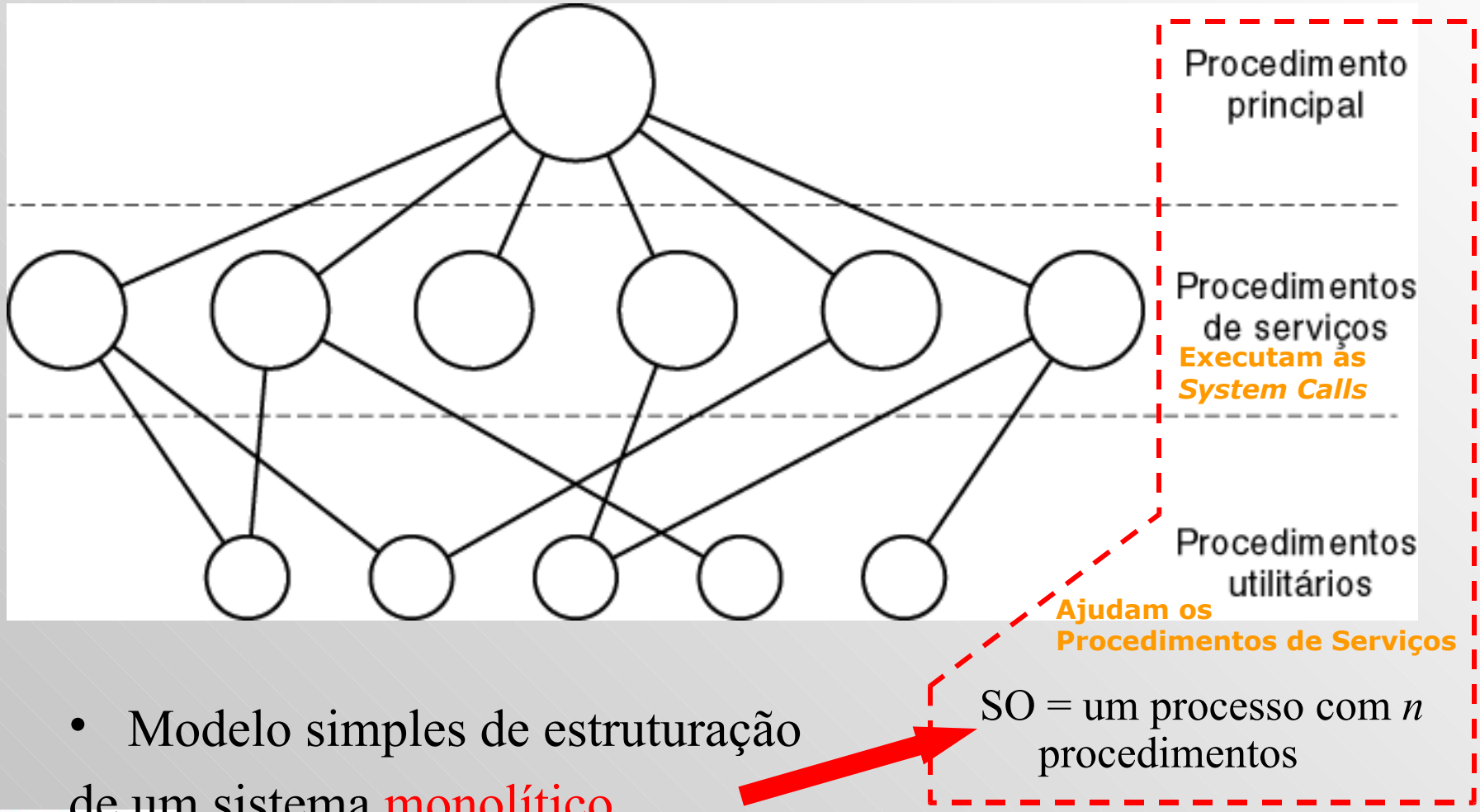
Estrutura de Sistemas Operacionais: Sistema Monolítico



- Modelo simples de estruturação de um sistema monolítico

SO = um processo com *n* procedimentos

Estrutura de Sistemas Operacionais: Sistema Monolítico



- Modelo simples de estruturação de um sistema monolítico

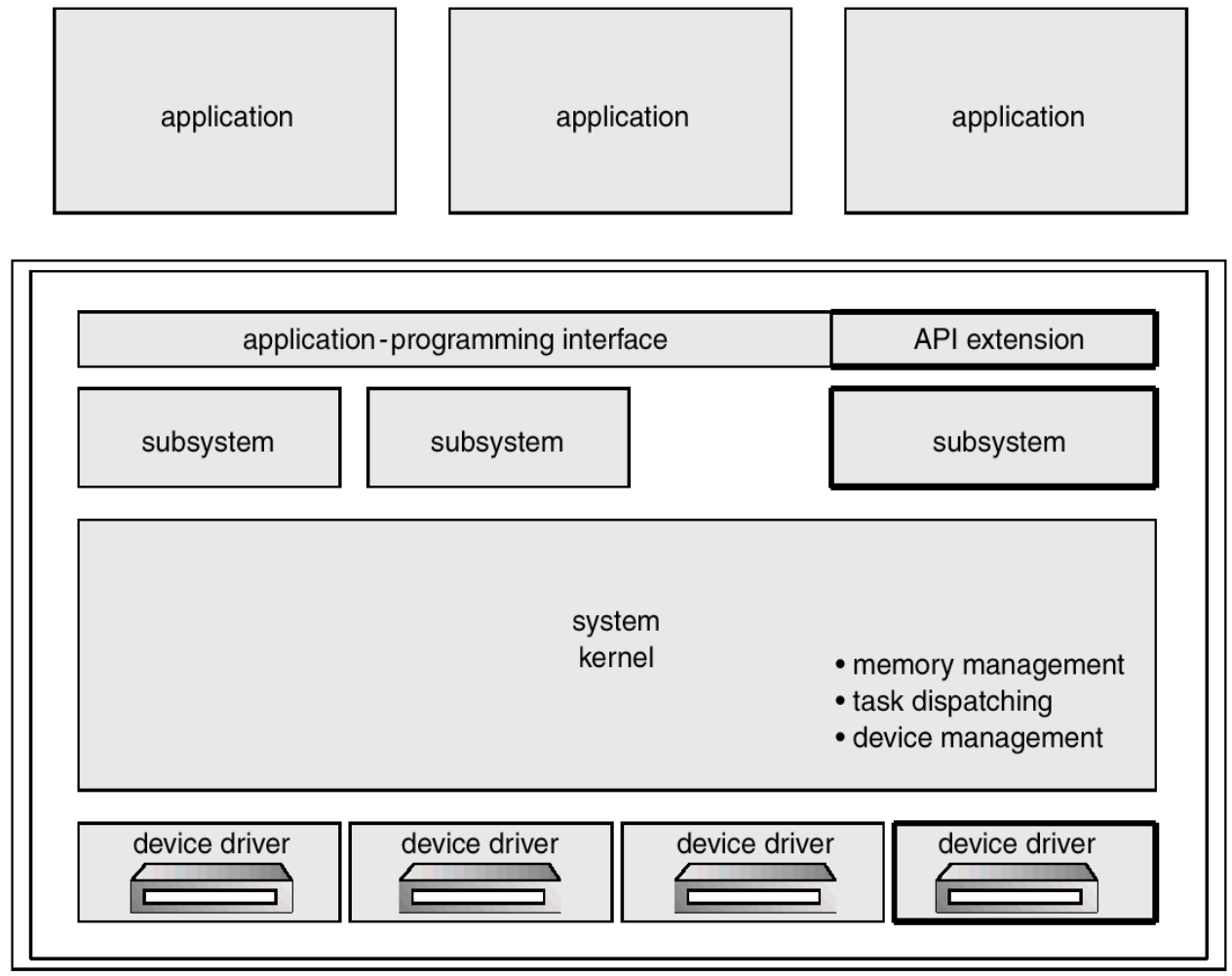
Estrutura de Sistemas Operacionais: Sistema em Camadas

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

- Modularidade
- Hierarquia
- Facilita evolução e adaptação a novos ambientes (Flexibilidade)

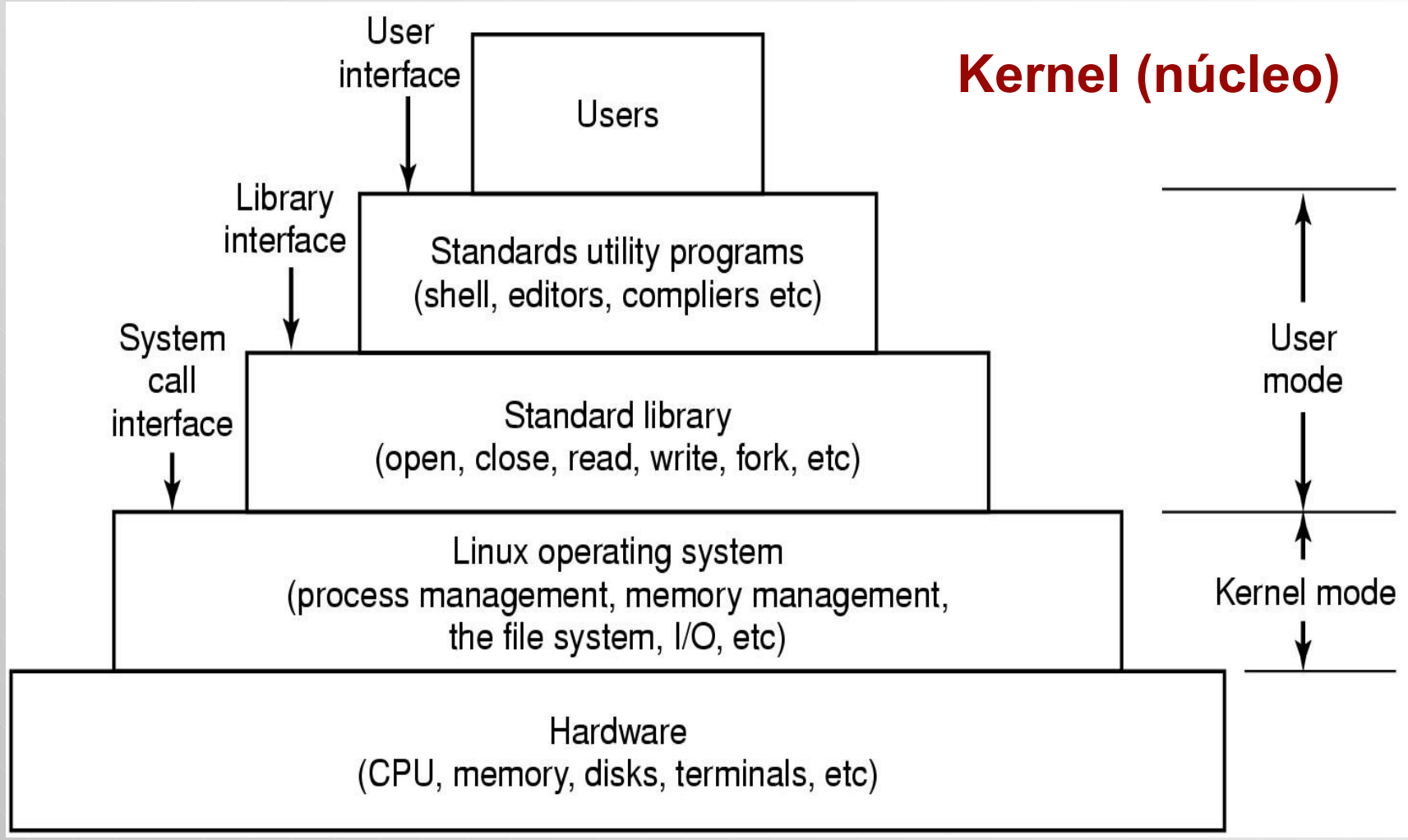
- Estrutura em camadas

Estrutura de Sistemas Operacionais: Sistema em Camadas

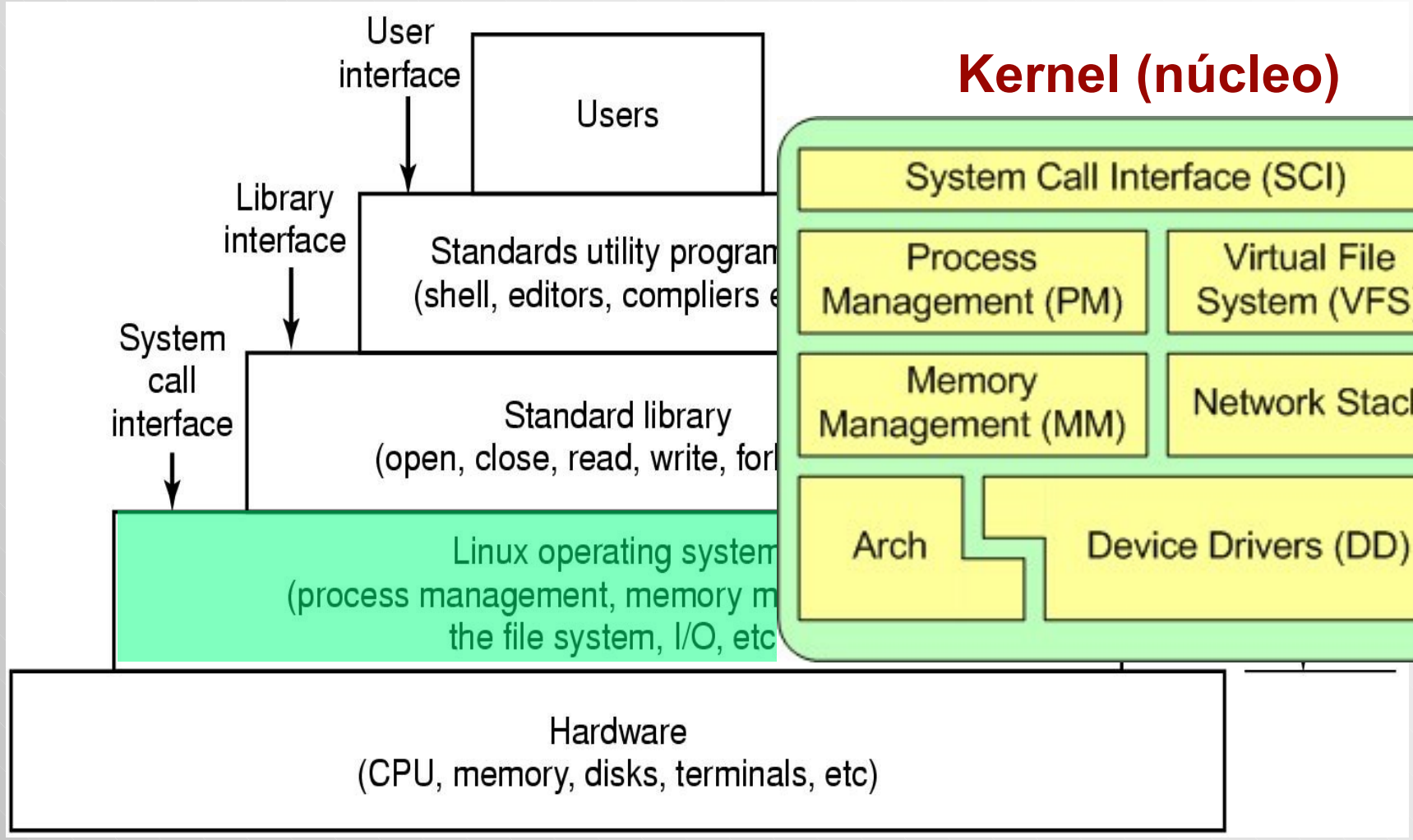


- Modularidade
- Hierarquia
- Facilita evolução e adaptação a novos ambientes (Flexibilidade)

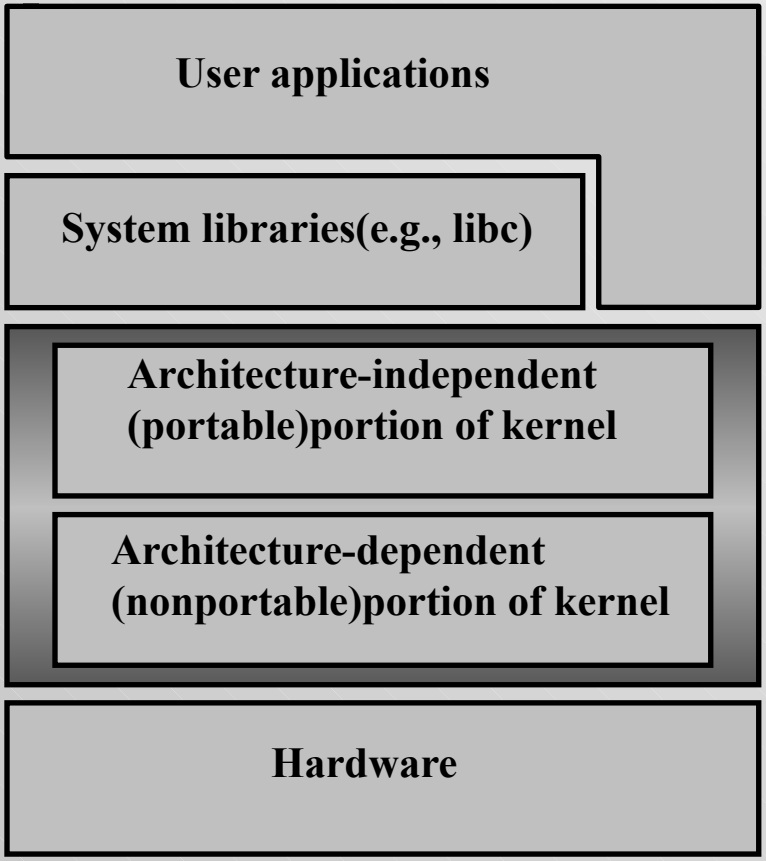
Camadas em Linux

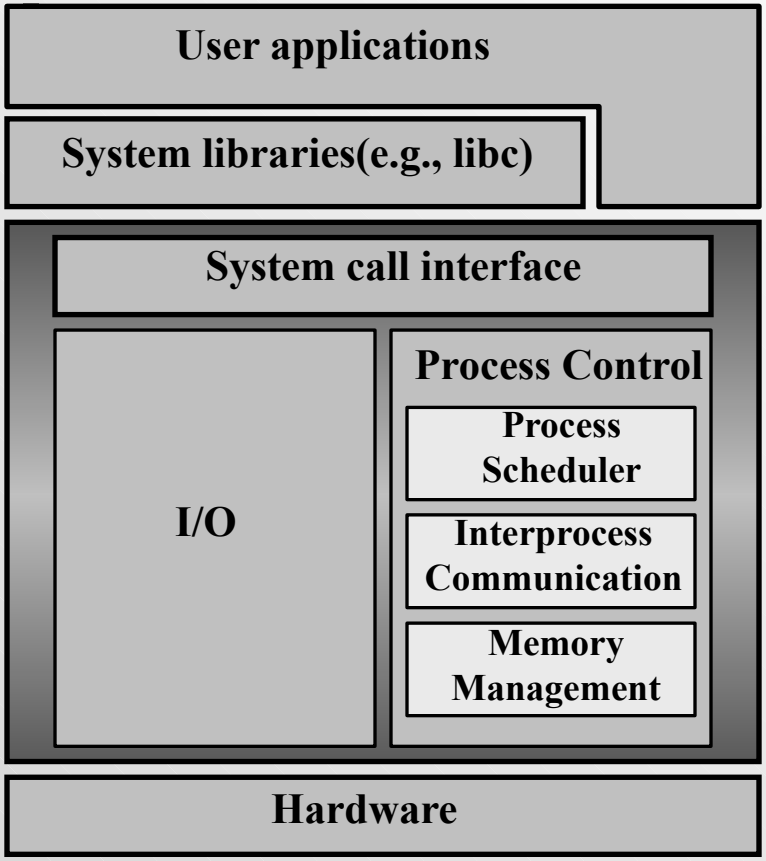


Camadas em Linux

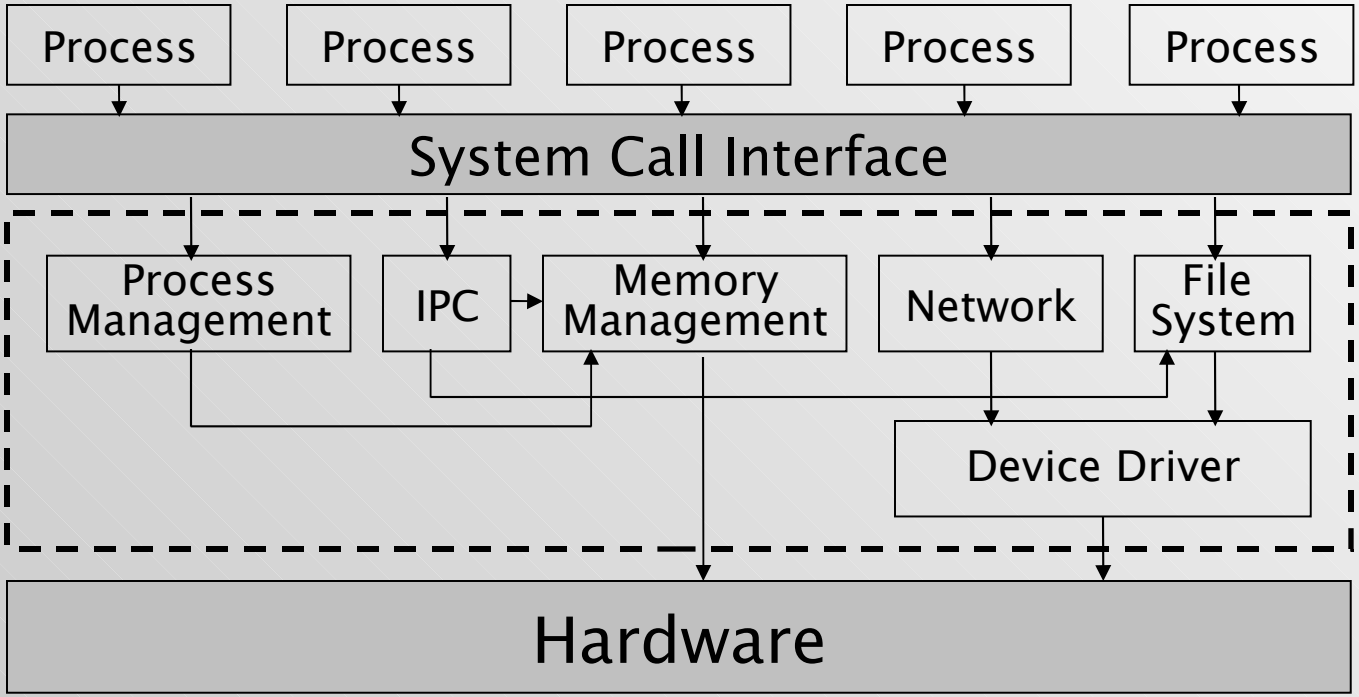


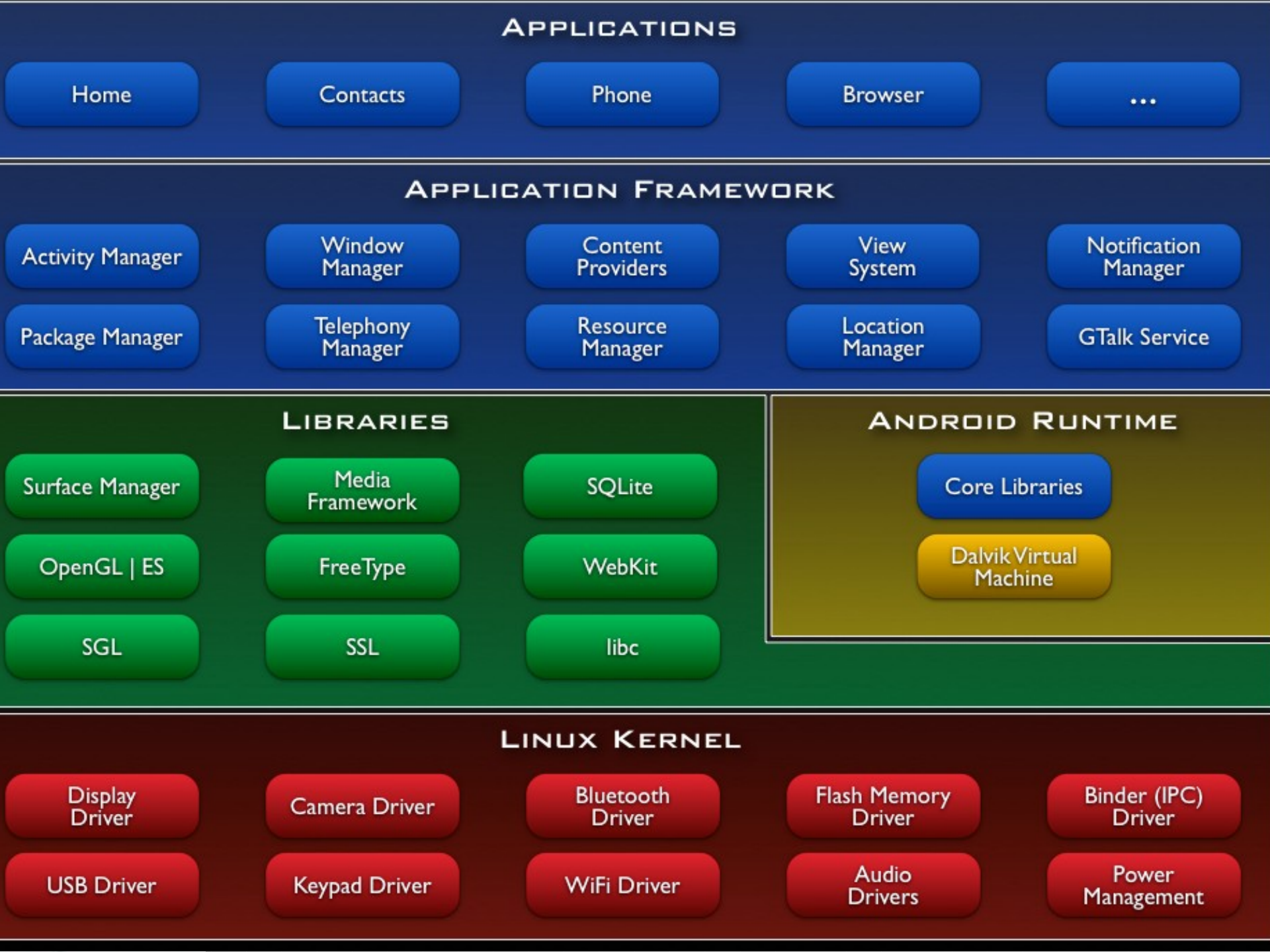
Arquitetura Linux - Outra Visão





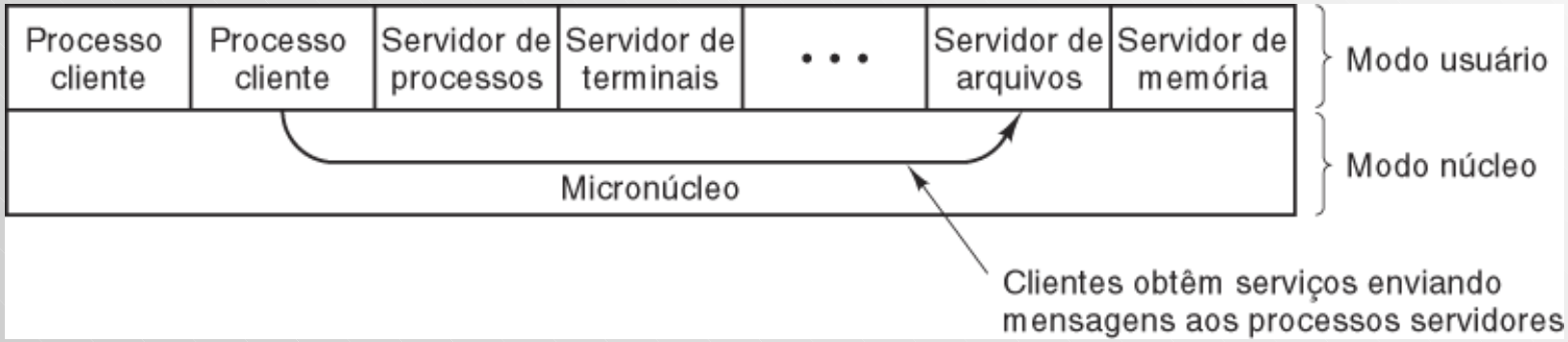
Arquitetura do Kernel do Linux




Applications
Home
Contacts
Phone
Browser
...
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource Manager
Location Manager
GTalk Service
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Camera Driver
Bluetooth Driver
Flash Memory Driver
Binder (IPC) Driver
USB Driver
Keypad Driver
WiFi Driver
Audio Drivers
Power Management

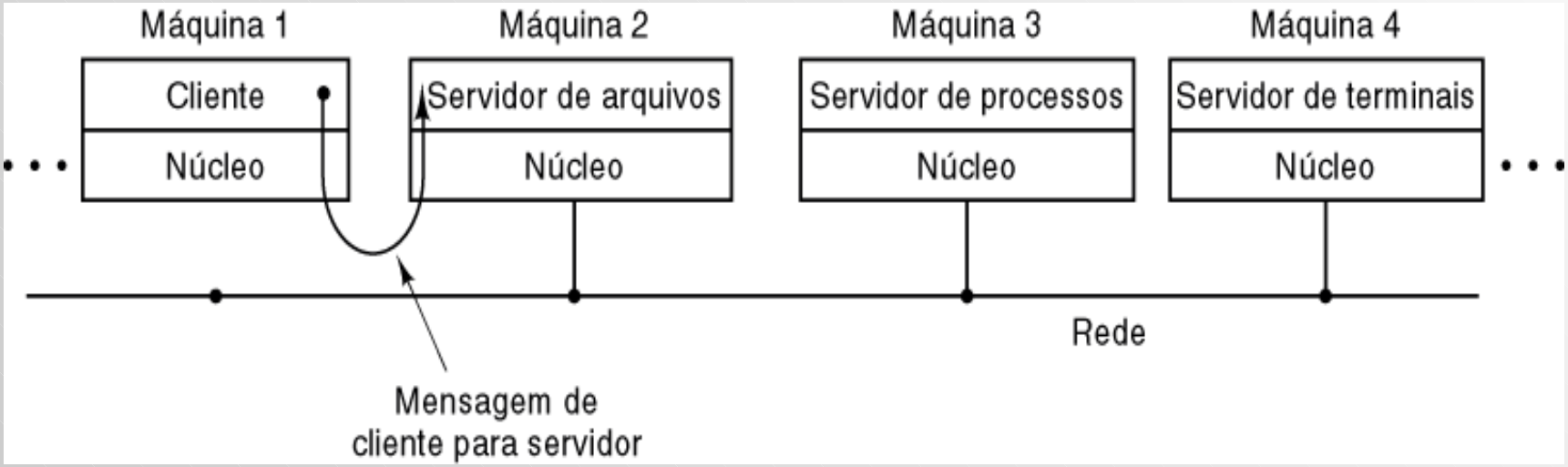
Android em Camadas

Estrutura de Sistemas Operacionais: Cliente-Servidor



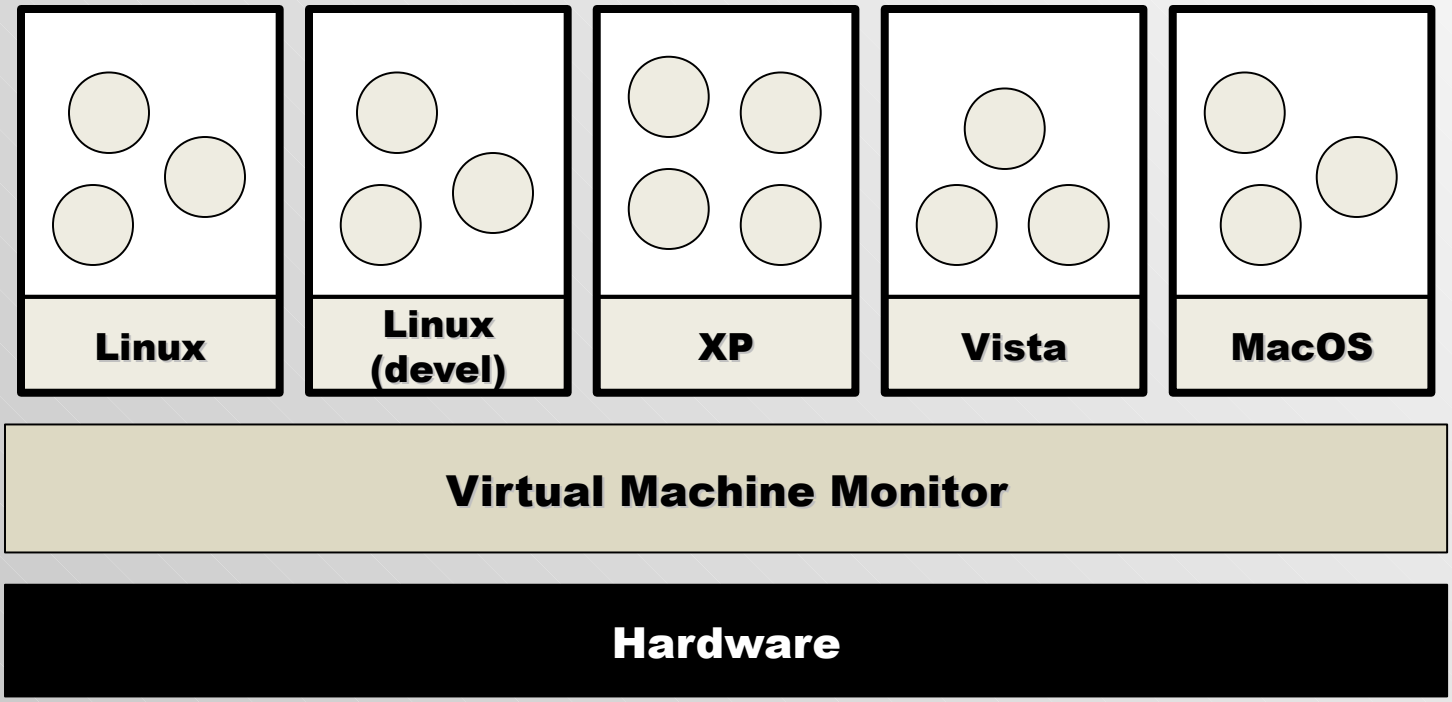
- O modelo cliente-servidor

Estrutura de Sistemas Operacionais: Cliente-Servidor (2)

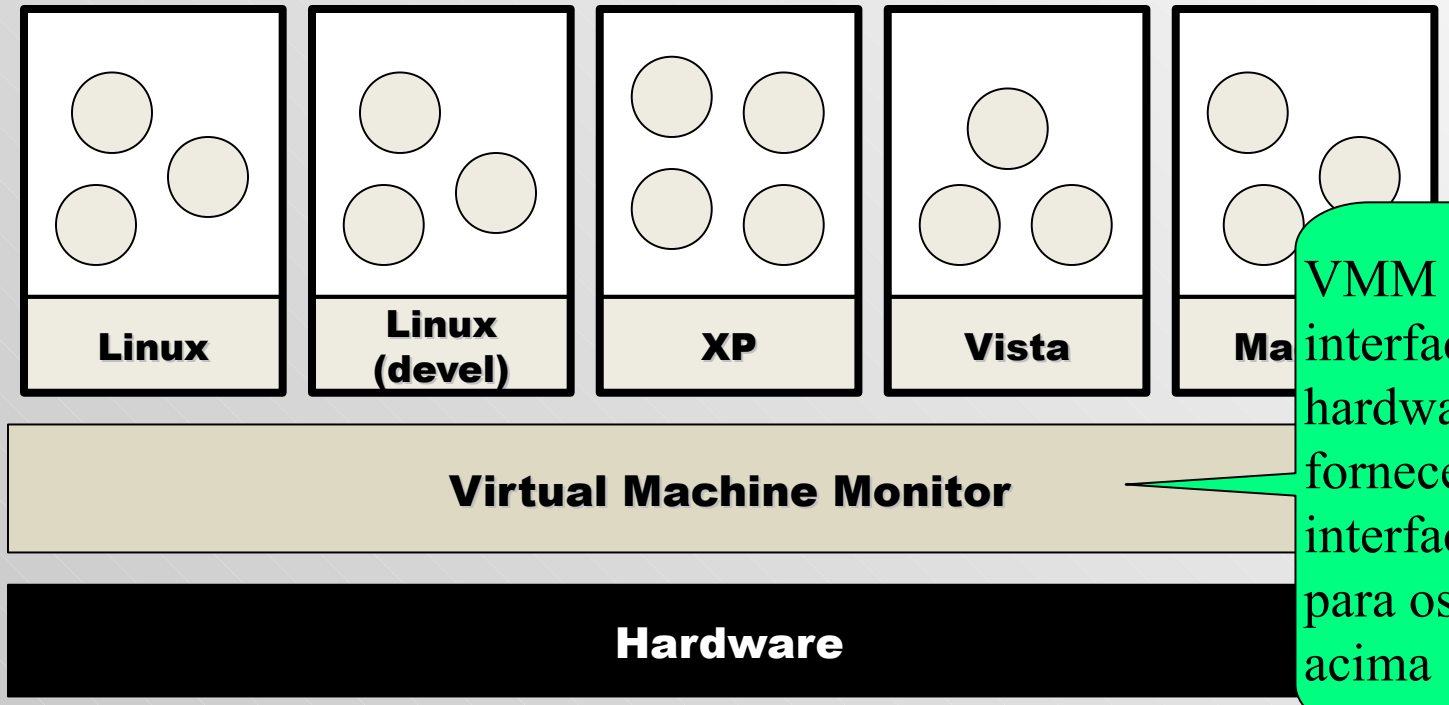


- O modelo cliente-servidor em um sistema distribuído

Estrutura de Sistemas Operacionais: Máquina Virtual (Virtualização)



Estrutura de Sistemas Operacionais: Máquina Virtual (Virtualização)

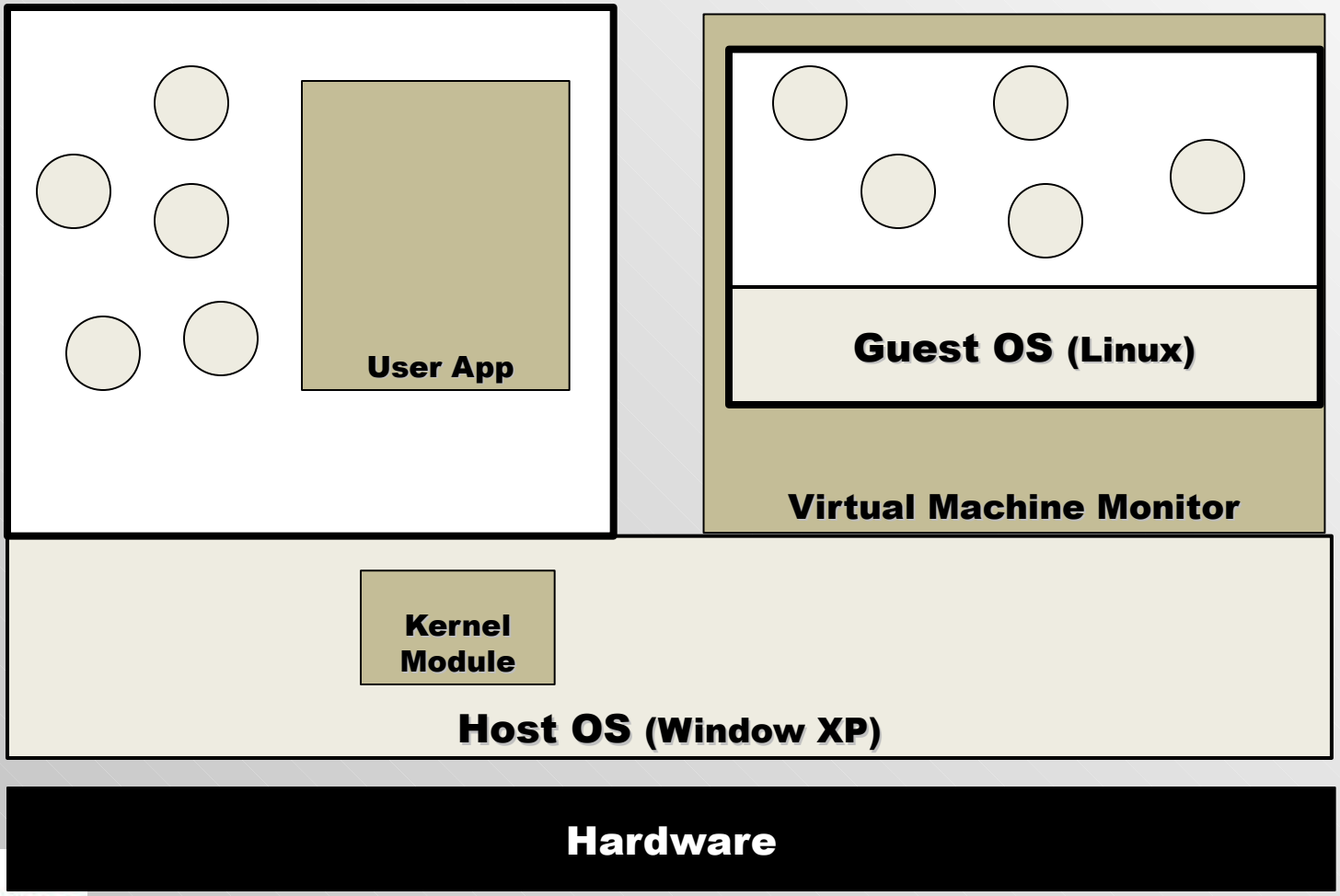


Tipos de Virtualização

- De processo
  - Java, .NET
- De dispositivo
  - RAID
- De sistema
  - VMware

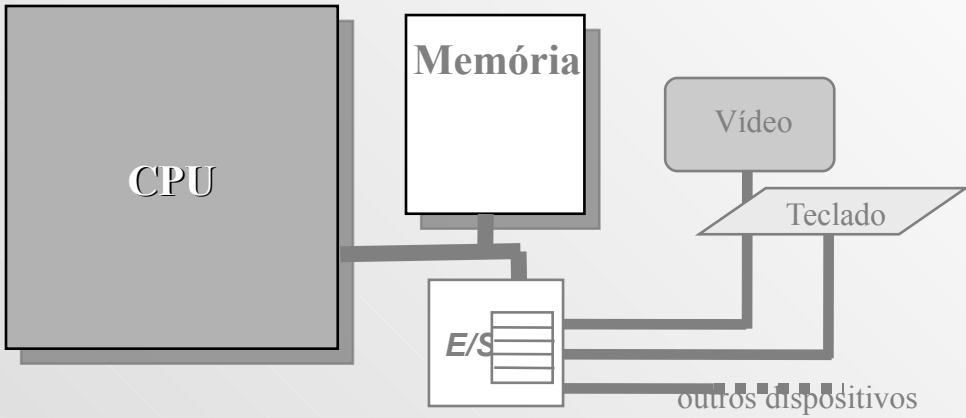
Arquitetura Hosted Monitor

- Executa como uma aplicação em um SO existente



Conceito Básico: Processo
(um programa em execução)

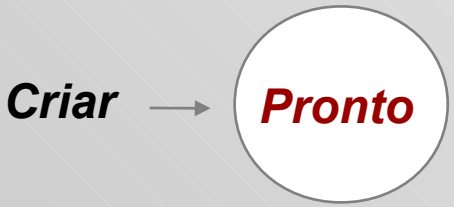
- Contexto de processo



- CPU: Registradores
- Memória: Posições em uso
  - Dentro de seu espaço de endereçamento
- E/S: Estado das requisições
- Estado do processo: Rodando, Bloqueado, Pronto
- Outras informações

Processo

- Estados de um processo



Contexto

ID do Processo
Estado
Prioridade
Program Counter
Ponteiros da Memória
Contexto (outros regs.)
I/O Status
Informações gerais • ***tempo de CPU*** • ***limites, usuário, etc.***

Processo

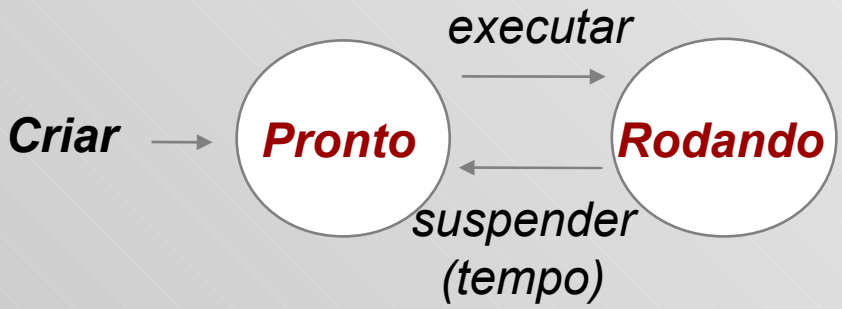
- Estados de um processo

Contexto

ID do Processo
Estado
Prioridade
Program Counter
Ponteiros da Memória
Contexto (outros regs.)
I/O Status
Informações gerais • ***tempo de CPU*** • ***limites, usuário, etc.***

Processo

- Estados de um processo

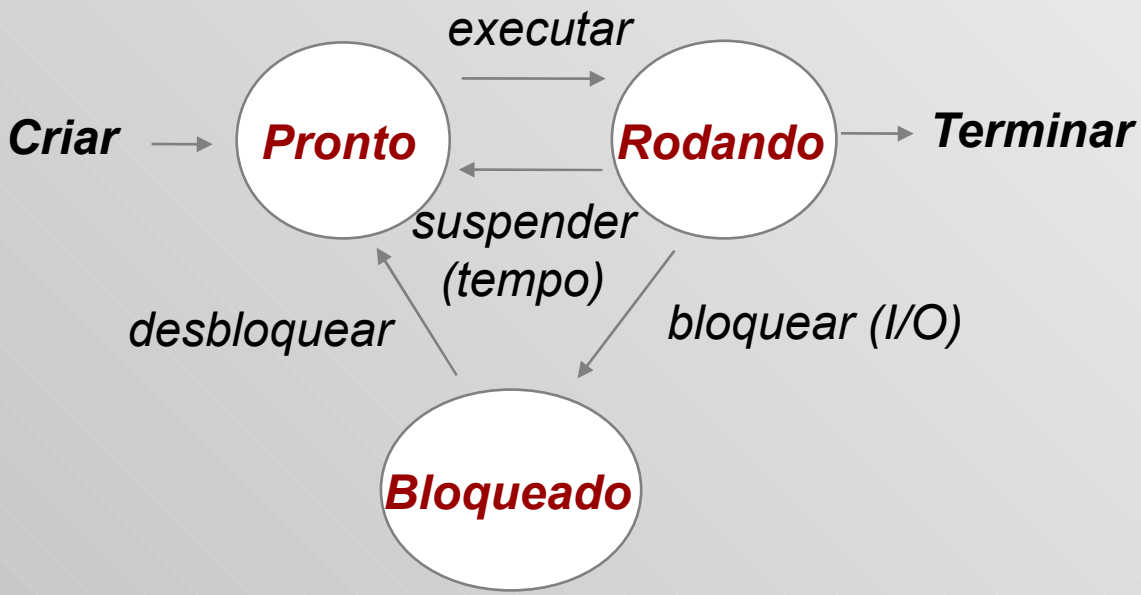


Contexto

ID do Processo
Estado
Prioridade
Program Counter
Ponteiros da Memória
Contexto (outros regs.)
I/O Status
Informações gerais • ***tempo de CPU*** • ***limites, usuário, etc.***

Processo

- Estados de um processo



Contexto

ID do Processo
Estado
Prioridade
Program Counter
Ponteiros da Memória
Contexto (outros regs.)
I/O Status
Informações gerais • ***tempo de CPU*** • ***limites, usuário, etc.***

Criação de Processos

- Principais eventos que levam à criação de processos
  - Início do sistema
  - Execução de chamada ao sistema de criação de processos
  - Solicitação do usuário para criar um novo processo
  - Início de um *job* em lote

Término de Processos

- Condições que levam ao término de processos
  - Saída normal (voluntária)
  - Saída por erro (voluntária)
  - Erro fatal (involuntário)
  - Cancelamento por um outro processo (involuntário)

Hierarquias de Processos

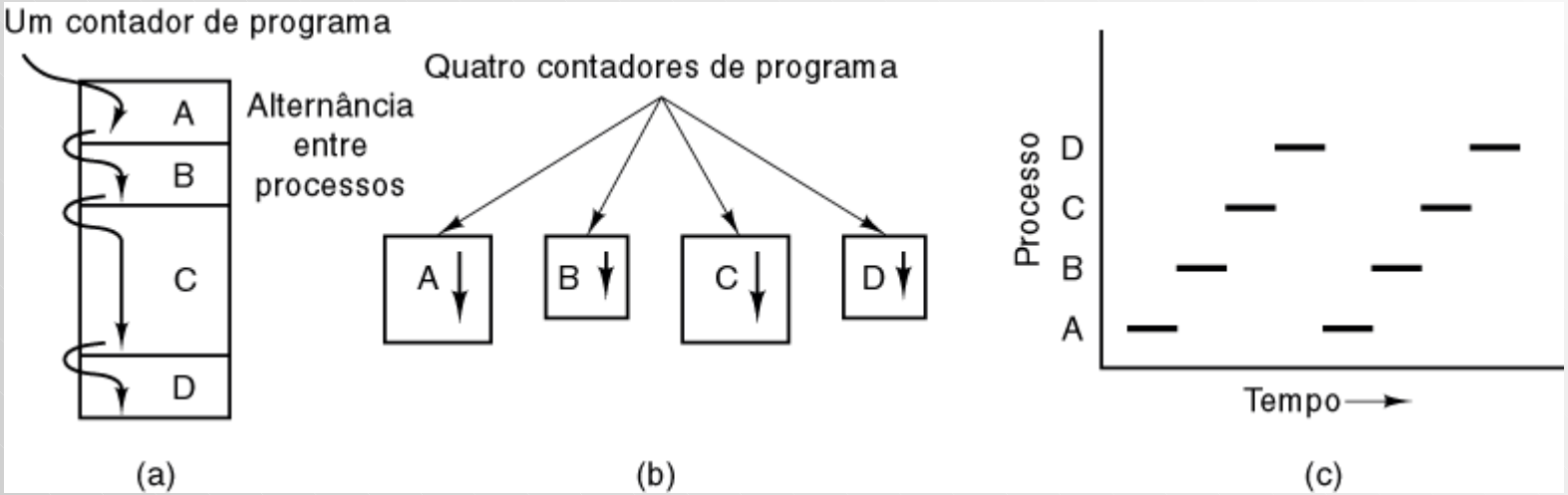
- Processo “pai” cria um processo “filho”, processo filho pode criar seu próprio processo …
- Formando uma hierarquia
  - UNIX chama isso de “grupo de processos”
- Windows não possui o conceito de hierarquia de processos
  - Todos os processos são criados iguais (sem conceito de “pai” e “filho”)

Exemplo de criação de processo

```
#include<stdio.h>

int main(int argc, char* argv[]) {
  int pid = fork();
  if (pid == 0) {
    for (int c = 0; c < 100000; c++ ) {
      printf("P1\n");
    }
  }
  else {
    for (int c = 0; c < 100000; c++ ) {
      printf("P2\n");
    }
  }
}
```

Conceito: Multiprogramação



a) Multiprogramação de quatro programas

a) Modelo conceitual de 4 processos sequenciais, independentes, mas

b) Somente um programa está ativo a cada momento => escalonamento

Escalonamento de processos

- Se há processos prontos para ser executados, o sistema operacional decide qual será executado primeiro
- A parte do sistema operacional responsável por essa decisão é chamada escalonador
    - O algoritmo usado para tal é chamado de algoritmo de escalonamento
- Para que um processo não execute tempo demais, praticamente todos os computadores possuem um mecanismo que periodicamente causa uma interrupção

O que vamos ver neste Módulo I?

- Gerenciamento de Processos / Escalonamento
- Gerenciamento de Entrada e Saída
- Gerenciamento de Memória / Memória Virtual
- Gerenciamento de Disco / Sistemas de Arquivos