UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Aula 01: Introdução a Sistemas Operacionais

Conceitos, estrutura e histórico

Prof. Rodrigo Campiolo
Prof. Rogério A. Gonçalves[1]

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Departamento de Computação (DACOM)
Campo Mourão, Paraná, Brasil

Ciência da Computação

BCC34G - Sistemas Operacionais

Sumário

Introdução a SO

Um computador moderno consiste em

- Um ou mais processadores.
- Memória principal.
- Discos.
- Diversos dispositivos de entrada e saída.

Necessidade

Para gerenciar todos esses componentes é necessária uma camada de software - o **Sistema Operacional**.

O que é um SO?

É uma camada de software que faz a interface entre o hardware, o usuário e os programas usuários.

- É uma estrutura ampla, complexa, que incorpora aspectos:
  - **baixo nível**, como *drivers* de dispositivos e gerência de memória física;
  - **alto nível**, como programas utilitários e a própria interface gráfica.

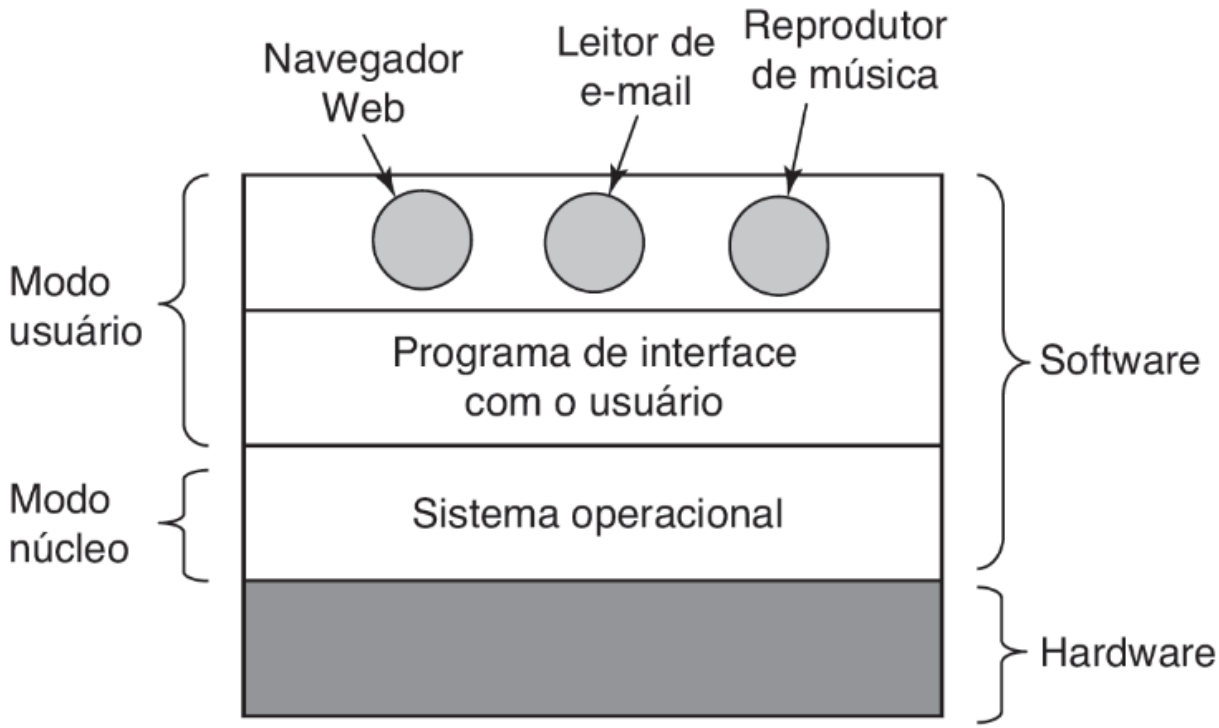


Figura 1: Localização do SO. *(Tanenbaum and Bos (2016))*

Introdução a SO

Definição

- É uma máquina estendida
  - Oculta os detalhes complicados que precisam ser executados
  - Apresenta ao usuário uma máquina virtual, mais fácil de usar
- É um gerenciador de recurso
  - Cada programa tem um tempo com o recurso
  - Cada programa tem um espaço no recurso

Definição: Abstração de Recursos

- O acesso aos recursos de hardware pode ser uma tarefa complexa, devido às características específicas dos dispositivos e a complexidade de suas interfaces.
- O SO pode prover mecanismos e interfaces abstratas:
  - Para o acesso aos dispositivos: acesso ao disco $\rightarrow$ primitivas (`open`, `read` e `close`).
  - Tornar os aplicativos independentes do hardware. (**portabilidade**)

Definição: Gerência de Recursos

- O SO define políticas para gerenciar o uso dos recursos de hardware pelos aplicativos, e resolver eventuais disputas e conflitos.
- **Recursos:**
  - Processador
  - Memória
  - Disco
  - Vídeo
  - Impressora
  - etc.

Definição: Gerência de Recursos

- Permite que múltiplos programas sejam executados ao mesmo tempo.
- Gerencia e protege a memória, os dispositivos de entrada e saída e outros recursos.
- Inclui a multiplexação (partilha) de recursos de duas maneiras diferentes:
  - No tempo
  - No espaço

Funcionalidades

- Gerência do Processador
- Gerência de Memória
- Gerência de Dispositivos (Entrada/Saída)
- Gerência de Arquivos
- Gerência de Proteção (Segurança)

Estrutura de um SO

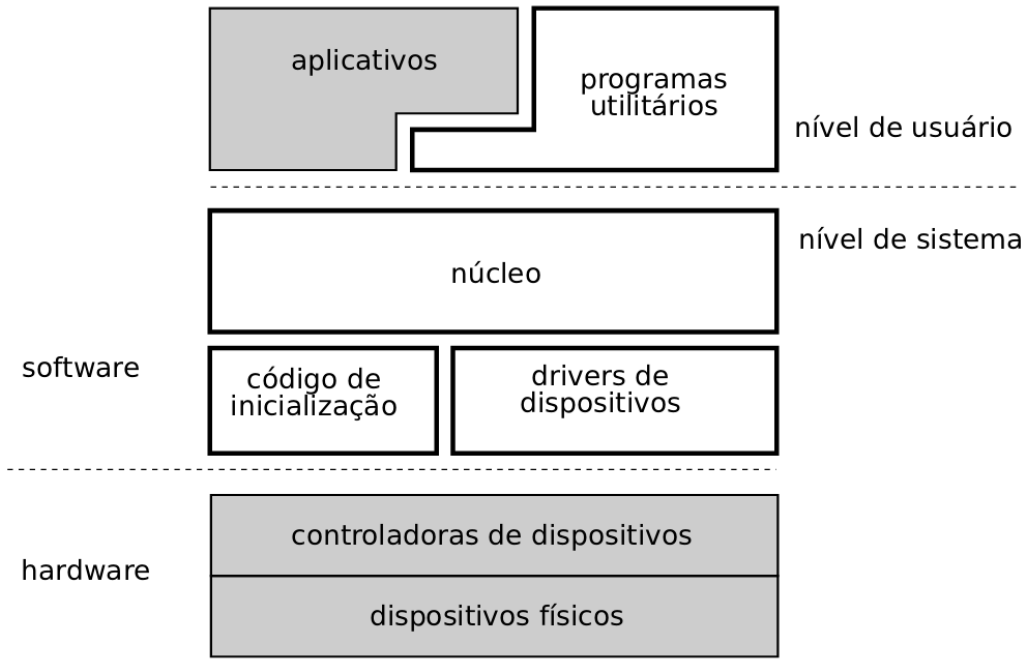


Figura 2: Visão geral da estrutura de um SO. *(Maziero (2017))*

Estrutura de um SO

- **Núcleo:** Ou *kernel* é o coração do SO, sendo o responsável pela gerência dos recursos do hardware, implementa as principais abstrações utilizadas pelos programas.
- **Drivers:** São módulos de código específicos para acessar os dispositivos físicos. Por exemplo: Driver da Placa de Vídeo.
- **Código de inicialização:** A inicialização do hardware requer uma série de tarefas complexas, como reconhecer e testar os dispositivos, e carregar o núcleo do sistema operacional em memória e iniciar sua execução.

Estrutura de um SO

- **Programas utilitários:** São programas que facilitam o uso do sistema computacional, fornecendo funcionalidades complementares ao núcleo.
- Como:
  - formatação de discos e mídias.
  - configuração de dispositivos.
  - manipulação de arquivos (mover, copiar, apagar).
  - interpretador de comandos.
  - terminal, interface gráfica, gerência de janelas, etc.
  - desenvolvimento de aplicações (tradutores, compiladores, linkers).

Interface Hardware-Software I

Interface Hardware-Software II

- O usuário interage com aplicativos desenvolvidos por programadores que fazem uso de utilitários tais como editores, compiladores, interpretadores, montadores etc. . .
- Por baixo destes aplicativos e utilitários pode haver uma interface amigável.
- Esta interface atuará sobre o sistema operacional (shell + kernel) que por sua vez utilizará recursos da BIOS.
- Todas estas camadas por sua vez serão de fato executadas pela unidade de controle (microprograma).
- Por último, estão os dispositivos físicos de entrada e saída, registradores e memória.

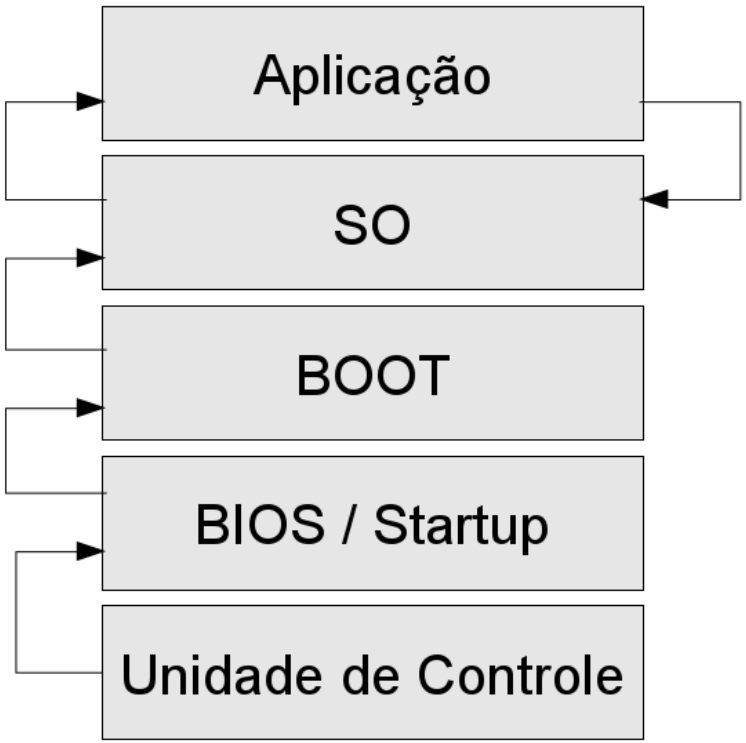
Carregamento e Transferência de Controle I

- Sabemos que os programas (processos) tem suas instruções executadas pelo processador.
- O Sistema Operacional (SO) é responsável por carregar os programas para a memória e gerenciar sua execução.
- Contudo, o SO também é um programa que precisa ser carregado na memória para entrar em execução.
- Solução: Um pequeno programa chamado *bootstrap ou boot* é armazenado no início da memória como ROM.
- Quando o computador é ligado, quem entra em execução é o Sistema Básico de Entrada/Saída (BIOS), que é um programa pré-gravado em um chip de memória ROM (firmware), sendo responsável pelo suporte básico de acesso ao hardware.

Carregamento e Transferência de Controle II

- Junto com o BIOS existem na ROM outros dois programas: Setup (configurar parâmetros do BIOS) e o POST (Power On Self Test ou Auto teste) que realiza uma sequência de testes e verificações do hardware.
- Após as verificações do BIOS, ele efetua a descompactação para a Memória RAM do *bootstrap*.
- O contador de programa (PC) da CPU é direcionado para o endereço da primeira instrução do *boot* executando as instruções deste programa que é responsável por carregar o SO ou parte dele que for necessária para iniciar o computador.
- Ao terminar de carregar o SO, o PC é redirecionado para a primeira instrução do SO na RAM.
- E então o SO está em execução.

Carregamento e Transferência de Controle III



Sequência

1. O hardware executa o Startup, que já está em posição pré-definida da memória ROM.
2. O Startup busca o BOOT que deve estar em uma posição pré-definida do disco e transfere o controle para ele.
   - O BIOS lê o setor zero (que contém apenas 512 bytes, denominado Master Boot Record) do HD.
   - Essa área contém um código que alavanca a inicialização (boot loader) do SO.
3. O BOOT LOADER carrega e inicializa o kernel e transfere o controle para o SO.
4. O SO busca e transfere o controle para a Aplicação.

Primeiros Sistemas Operacionais I

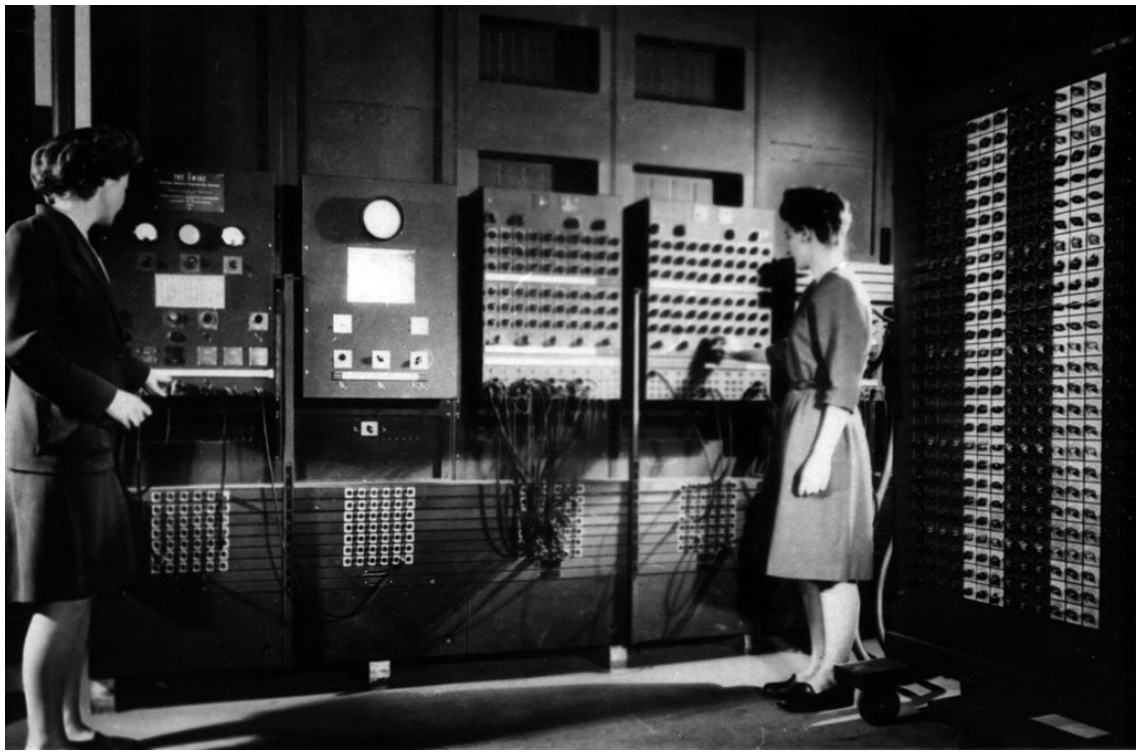
Gerações

- (1945-55) Válvulas: Não tinha software, circuitos.
- (1955-65) Transistores e sistemas em lotes (batch)
- (1965-1980) CIs e multiprogramação
- (1980-Presente) Computadores pessoais

Primeiros Sistemas Operacionais II

Computadores a Válvulas

- ENIAC: 1943-1946
- EDVAC: 1946-1952 (Neumann, conceito de programa armazenado)



Primeiros Sistemas Operacionais III

ATLAS: Universidade de Manchester

- De 1950 a 1960
- Processamento em Lotes
- Uso de Spooling
- Memória principal (98K)

Primeiros Sistemas Operacionais IV

CTSS: Projetado pelo MIT

- +-1962
- SO de tempo compartilhado
- Usado no IBM7090 (MEM 32K)
- 5K p/ SO e 27K p/ usuário
- Permitia processos concorrentes
- Sistema bem sucedido $\rightarrow$ MULTICS

OS/360: da IBM

- 1964
- Para a família IBM/360, desde pequenas máquinas comerciais até grandes máquinas científicas (mainframes).
- Conjunto único de programas para toda a família
- Facilitava manutenção e migração de programas entre máquinas da família.
- Escrito em linguagem de montagem
- Milhões de linhas de código
- Não executava nada muito bem

Primeiros Sistemas Operacionais VI

MULTICS: MIT, GE e Bell Labs.

- 1965
- Tempo Compartilhado
- Sistema de arquivos compartilhado
- Uso de paginação e segmentação
- 300 mil linhas de código
- Evoluiu para o UNICS $\rightarrow$ UNIX

Primeiros Sistemas Operacionais VII

XDS-940: Universidade da Califórnia (Berkeley)

- 1965
- Processamento de tempo compartilhado
- Memória de usuário (16K)
- Memória principal (64K)

Primeiros Sistemas Operacionais VIII

THE: Technische Hogeschool (Holanda)

- 1967
- Processamento em lote
- Memória de usuário (32K)
- Programas escritos em ALGOL
- Memória secundária (512K)

Primeiros Sistemas Operacionais IX

RC 4000: Regnecentralen

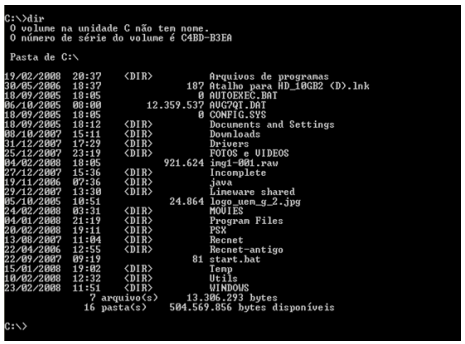
- 1970
- SO fortemente centrado no núcleo
- Permitia processos concorrentes
- Provia troca de mensagens

Evolução dos SOs - 4ª Geração I

- **Computadores Pessoais**
- Com a tecnologia de circuitos integrados de larga escala (LSI) surgem chips com milhares de transistores encapsulados em um centímetro quadrado de silício
  - Intel - 8080
  - IBM - PC
  - Apple - Macintosh

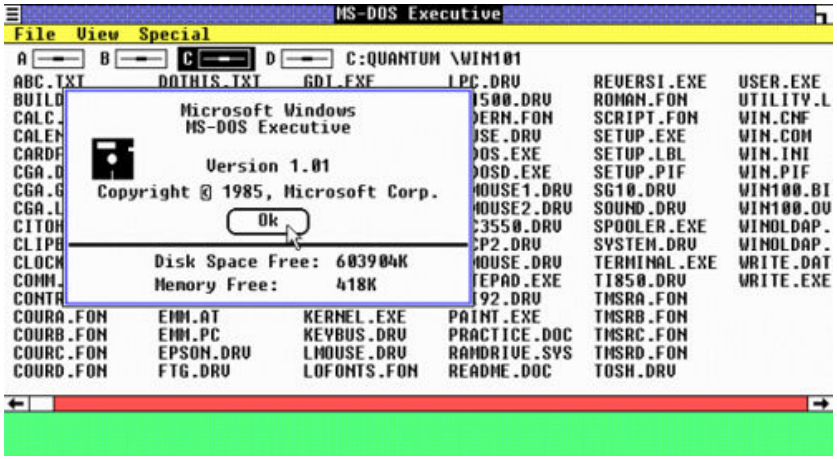
Evolução dos SOs - 4ª Geração II

- **Intel 8080 - CP/M** (Control Program/Monitor ou Control Program for Microcomputers) da Digital Research (Gary Kildall)
  - Sistema operacional baseado em disco;
- **IBM PC - DOS**
  - Inicialmente, a IBM tentou utilizar o CP/M, mas Kildall não quis nenhum acordo;
  - IBM procurou Bill Gates pedindo um sistema operacional para ser vendido juntamente com o IBM PC;
  - (1981) Bill Gates comprou a empresa que desenvolvia o DOS (Disk Operating System): *Seattle Computer Products*; Desenvolvedor: Tim Paterson;



Evolução dos SOs - 4ª Geração III

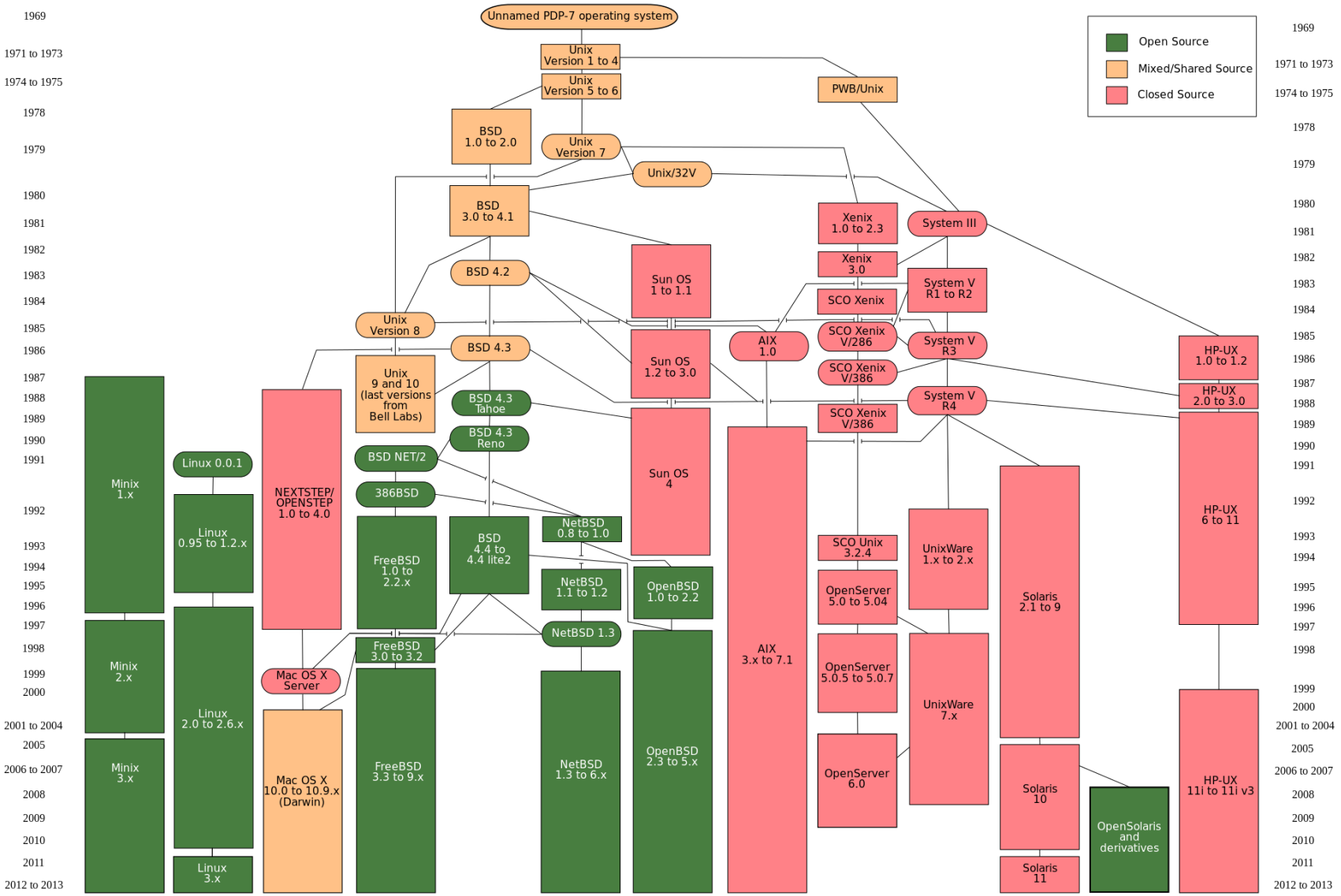
- Evolução do DOS → MS-DOS (Microsoft DOS)
- Tanto o CP/M quanto o MS-DOS eram baseados em comandos.
- Macintosh Apple - Sistemas baseados em janelas (GUI - Graphical User Interface)
- Microsoft - Plataforma Windows



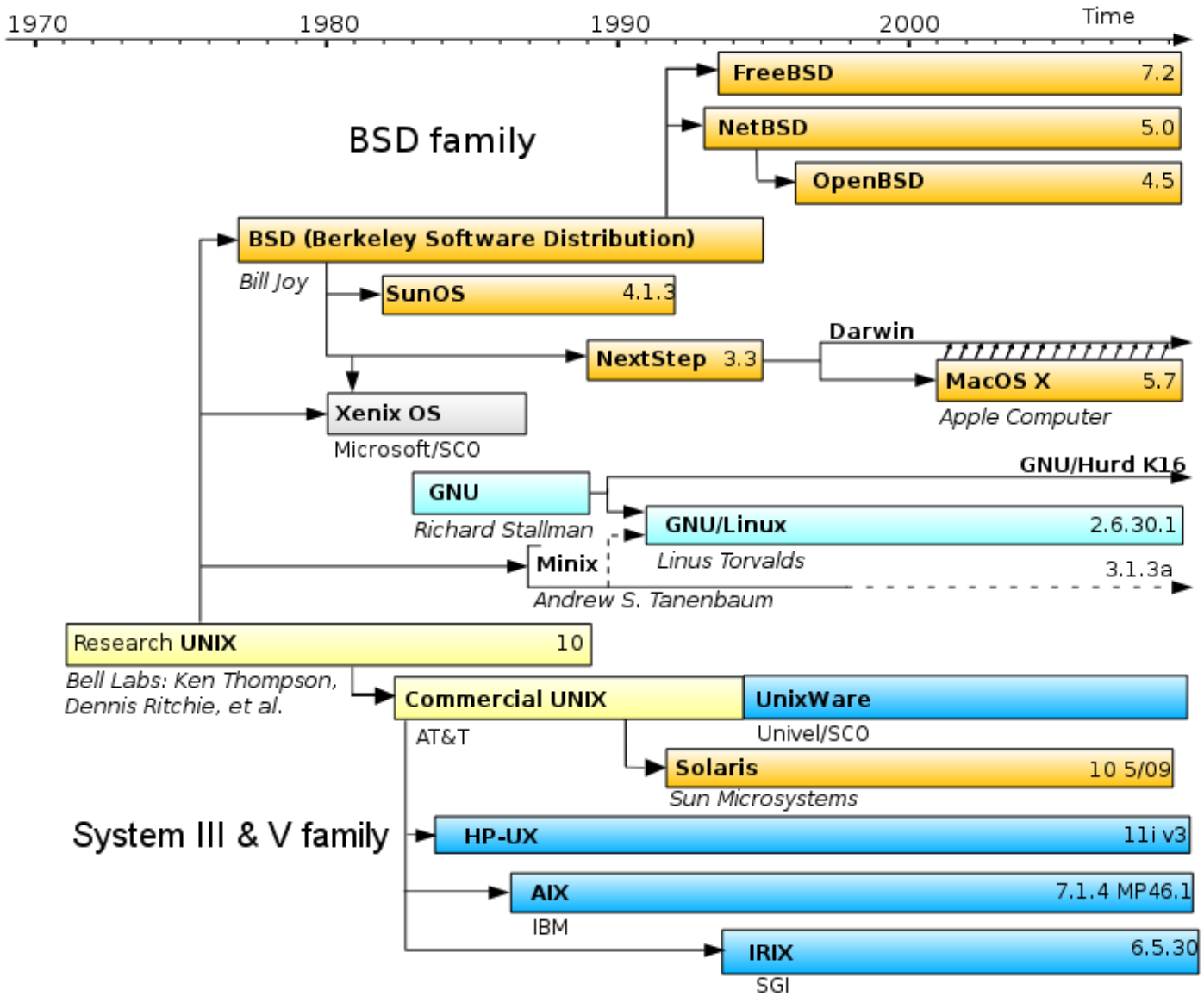
Evolução dos SOs - 5ª Geração

- **Era da computação distribuída:** um processo é dividido em subprocessos que executam em sistemas multiprocessados e em redes de computadores ou até mesmo em sistemas virtualmente paralelos
  - Sistemas Operacionais em Rede;
  - Linux;
  - Família Windows (NT, 95, 98, 2000, 2003 Server, XP, Vista, 7, 8);
  - Sistemas Operacionais Embarcados (Windows Mobile, TinyOS, AndroidOS, SymbianOS, PalmOS);
  - Sistemas Operacionais de Tempo Real;

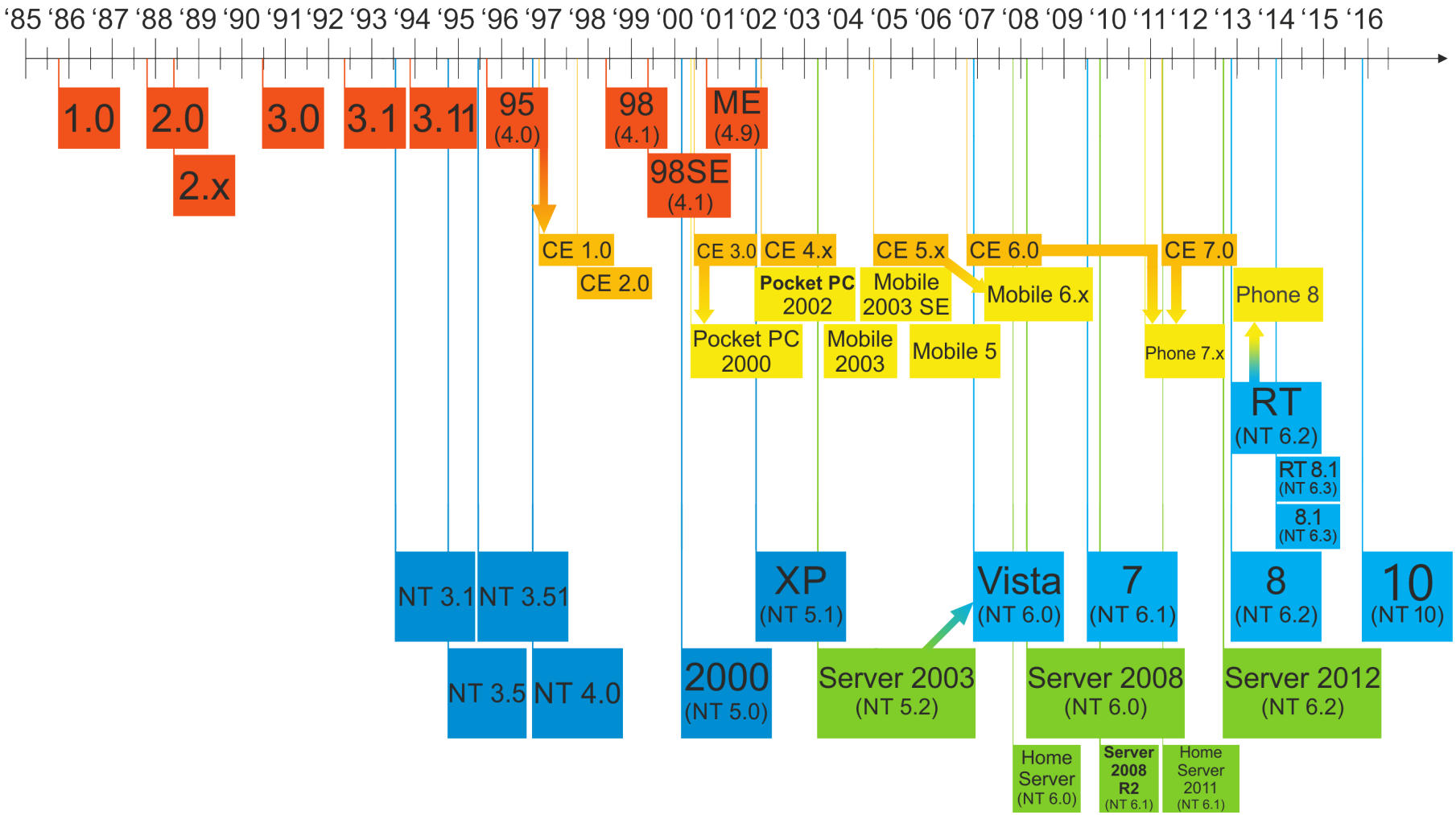
Histórico: Unix



Histórico: Unix



Histórico: Windows



Questões

1. Como as camadas superiores de software são executadas pelo hardware?
2. O hardware consegue distinguir se as instruções são do SO ou se é de uma aplicação qualquer?
3. Como as instruções de um programa são colocadas na memória, para serem executadas pelo hardware?
4. Um programa compilado para um tipo de arquitetura pode ser executado em outro tipo de arquitetura?
5. Depois que um programa fonte foi compilado e está pronto para ser executado, quem o coloca na memória?
6. Mas quem coloca o S.O. na memória?
7. Mas quem carrega o BOOT na memória?
8. Mas onde estão a BIOS e o Start Up?

Atividades

1. Fazer um resumo com o histórico dos Sistemas Operacionais.

Leitura Recomendada

1. Capítulo de Introdução do Livro de Sistemas Operacionais.

Referências

Deitel, H. M., Deitel, P. J., and Choffnes, D. R. (2003). *Operating systems*. Prentice-Hall, Inc., 3rd edition.

Maziero, C. A. (2017). *Sistemas operacionais: conceitos e mecanismos*. online. Disponível em http://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=so:so-livro.pdf.

Silberschatz, A., Galvin, P. B., and Gagne, G. (2015). *Fundamentos de sistemas operacionais*. LTC, 9 edition.

Stallings, W. (2012). *Operating systems: internals and design principles*. Pearson Education, 7th edition.

Tanenbaum, A. S. and Bos, H. (2016). *Sistemas operacionais modernos*. Pearson Education do Brasil, 4 edition.