Sistemas Operacionais

Prof. Jó Ueyama

Apresentação baseada nos slides da Profa. Dra. Kalinka Castelo Branco, do Prof. Dr. Antônio Carlos Sementille e nas transparências fornecidas no site de compra do livro “Sistemas Operacionais Modernos”

Aula de Hoje

1. **Introdução ao conceito de Sistemas Operacionais (SOs)**
2. **Histórico e evolução**

Aula de Hoje (conteúdo detalhado)

1. Introdução
1.1 Sistema Computacional
1.2 A importância dos SOs
1.3 Definição do SO
1.4 A interação com o SO
1.5 A evolução dos SOs

Introdução

1.1 Sistema Computacional

- **Consiste de:**
  - Um ou mais processadores
  - Memória principal
  - Discos, impressoras, teclado, monitor, interfaces de redes e outros dispositivos de entrada e saída

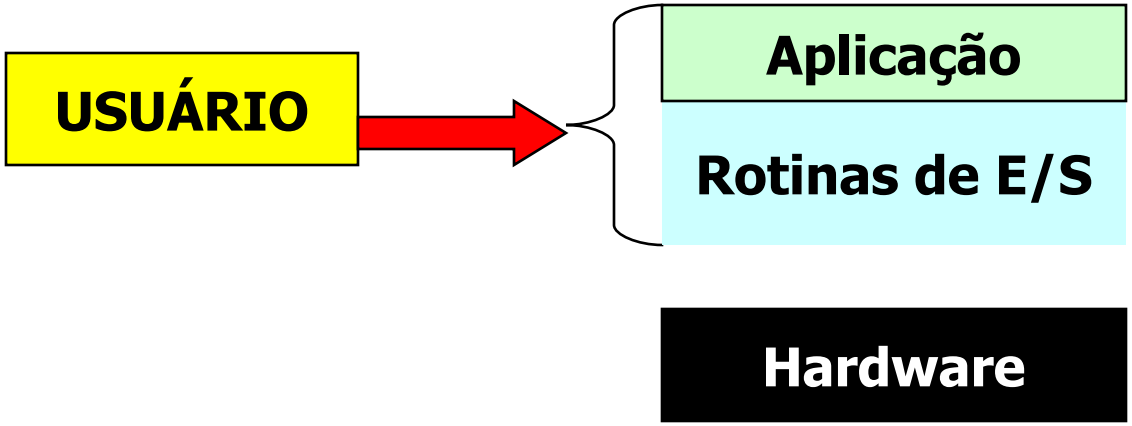
Aula de Hoje (conteúdo detalhado)

1. Introdução
1.1 Sistema Computacional
1.2 A importância do SOs
1.3 Definição do SO
1.4 A interação com o SO
1.5 A evolução do SOs

Introdução

1.2 A Importância do Sistema Operacional

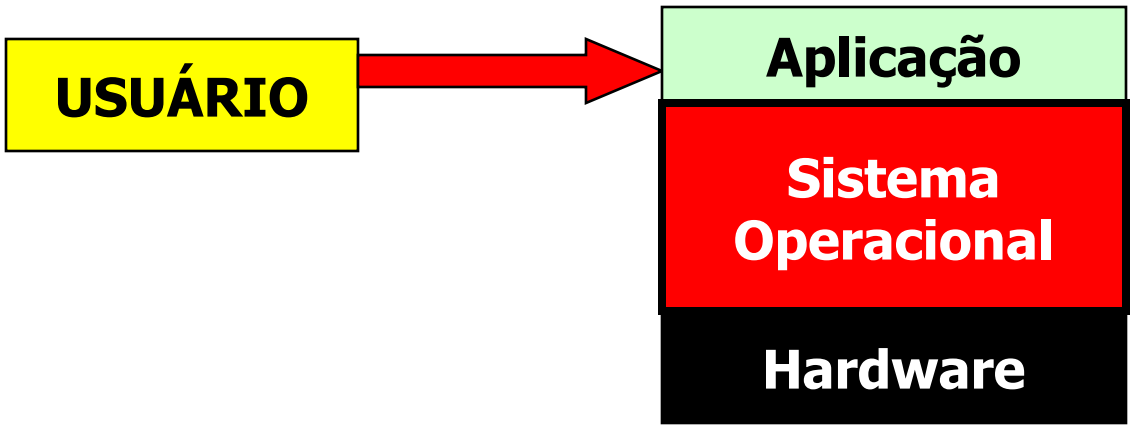
- **Sistema sem S.O.**
  - Gasto maior de tempo de programação
  - Aumento da dificuldade
  - Usuário preocupado com detalhes de hardware



Introdução

1.2 A Importância do Sistema Operacional

- **Sistema com S.O.**
  - Maior racionalidade (*separation of concerns*)
  - Maior dedicação aos problemas de alto nível
  - Maior portabilidade (Por que?)



Máquinas Multinível

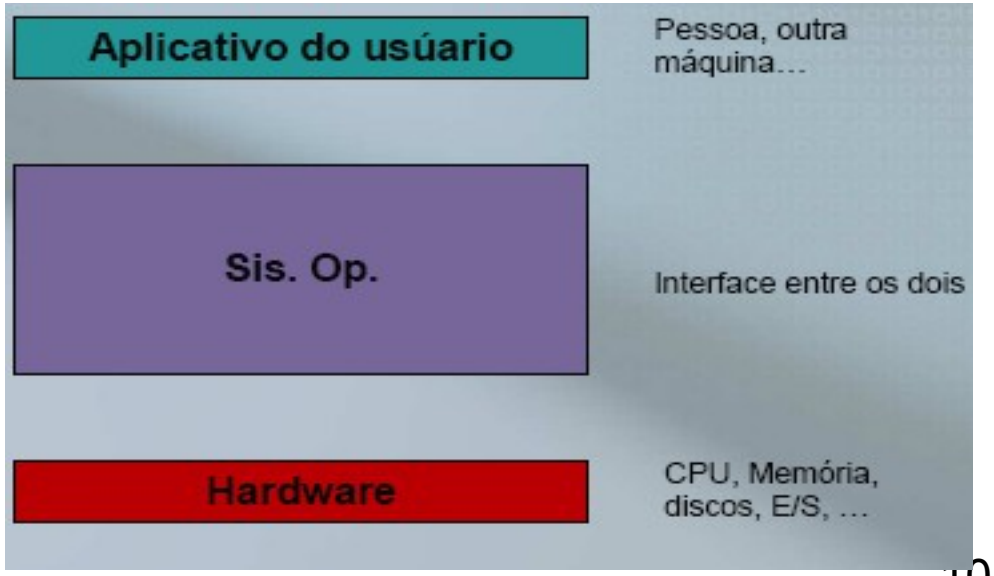
Aula de Hoje (conteúdo detalhado)

1. Introdução
1.1 Sistema Computacional
1.2 A importância do SOs
1.3 Definição do SO
1.4 A interação com o SO
1.5 A evolução do SOs

Introdução

1.3 Definição de Sistema Operacional

Um sistema operacional é um programa, ou conjunto de programas, interrelacionados cuja finalidade é agir como a) intermediário entre o usuário e o *hardware;* e b) gereciador de recursos.



Introdução

- O Sistema Operacional é uma interface HW/SW aplicativo
- Duas formas de vê-lo:
  - É um “fiscal” que controla os usuários
  - É um “juiz” que aloca os recursos entre os usuários

- Objetivos contraditórios:
  - Conveniência
  - Eficiência
  - Facilidade de evolução
  - A melhor escolha sempre DEPENDE de alguma coisa…

Introdução

- Possui várias vantagens, entre elas:
  - apresentar uma máquina mais flexível;
  - permitir o uso eficiente e controlado dos componentes de *hardware*;
  - permitir o uso compartilhado e protegido dos diversos componentes de *hardware* e *software*, por diversos usuários.

Introdução

- O Sis. Op. deve fornecer uma interface aos programas do usuário
  - Quais recursos de HW?
  - Qual seu uso?
  - Tem algum problema? (Segurança, falha…?)
  - É preciso de manutenção?
  - Chegou um email?
  - Entre outros…
  - Chamadas de sistema [e.g. *malloc*()] – programas de sistema
    - Chamada de alguma funcionalidade implementada no núcleo do SO

Aula de Hoje (conteúdo detalhado)

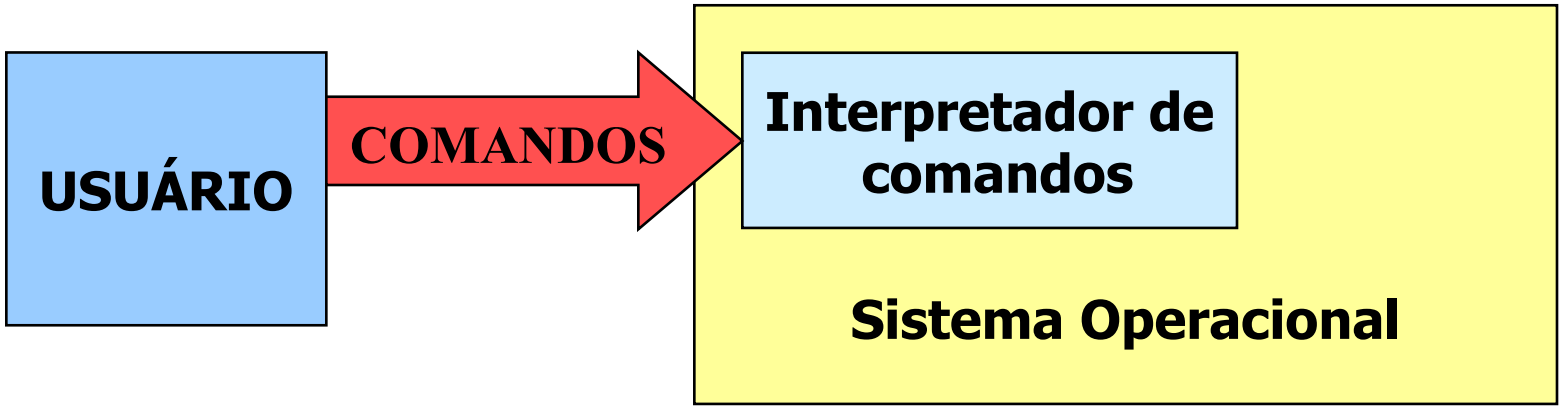
1. Introdução
1.1 Sistema Computacional
1.2 A importância do SOs
1.3 Definição do SO
1.4 A interação com o SO
1.5 A evolução do SOs

Introdução

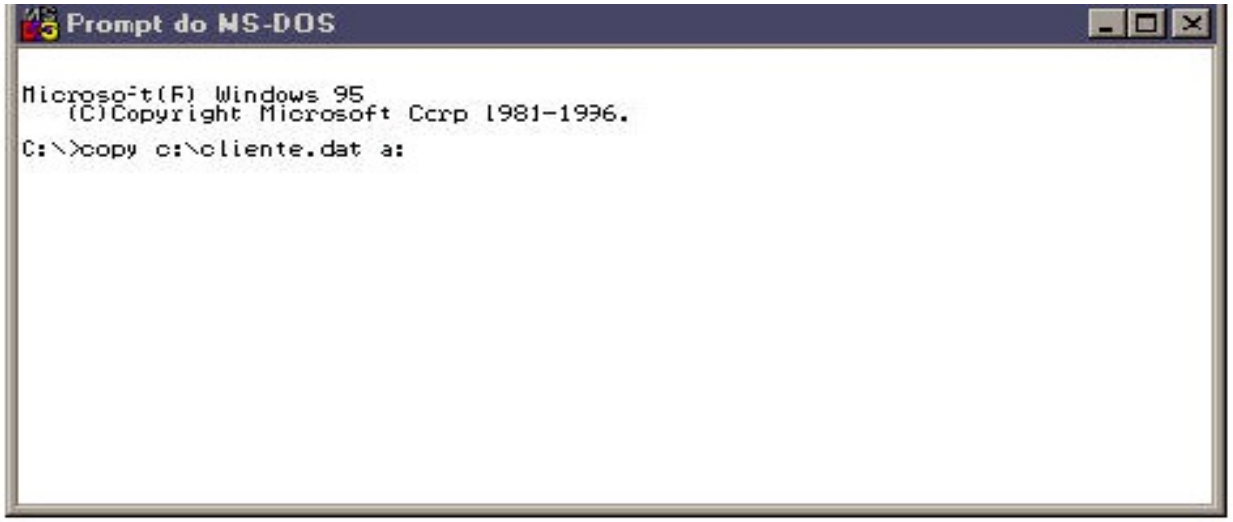
1.4 Interação com o Sistema Operacional

- **O USUÁRIO**

Interage com o S.O. de maneira direta, através de comandos pertencentes a uma linguagem de comunicação especial, chamada "linguagem de comando".
Ex: JCL (Job Control Language), DCL (Digital Control Language),...

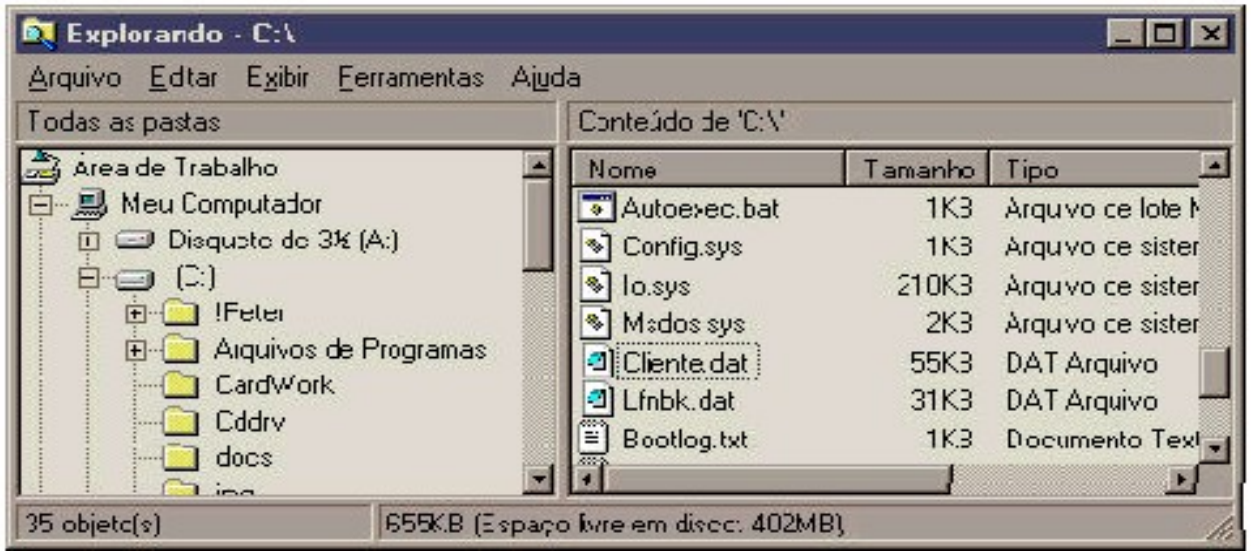


Introdução



Interface em Modo Texto (Linha de Comando)

Interface Texto



Interface Gráfica (GUI)


Finder File Edit View Go Window Help
Fri 9:52 PM
Macintosh HD
Guest
DEVICES
Macintosh HD
iDisk
PLACES
Desktop
Guest
Applications
Documents
SEARCH FOR
Today
Yesterday
Past Week
All Images
All Movies
All Documents
Desktop
Documents
Downloads
Library
Movies
Music
Pictures
Public
Sites
9 items, 15.86 GB available

Mac OS

Introdução

1.4 Interação com o Sistema Operacional

- **OS PROGRAMAS DE USUÁRIO**

Invocam os serviços do S.O. por meio das "chamadas ao sistema operacional".

Aula de Hoje (conteúdo detalhado)

1. Introdução
1.1 Sistema Computacional
1.2 A importância do SOs
1.3 Definição do SO
1.4 A interação com o SO
1.5 A evolução do SOs

Introdução

1.5 A Evolução dos Sistemas Operacionais

- Um SO pode processar sua carga de trabalho de duas formas
  - Serial (recursos alocados a um único programa)
  - Concorrente (recursos dinamicamente re-associados entre uma coleção de programas em diferentes estágios)
- Alcance e extensão de serviços
  - Depende do ambiente em que devem suportar (e.g. *cut down Linux versions* em sensores)

Histórico

Geração Zero – Computadores Mecânicos (1642 - 1945)

- Blaise Pascal (1623 - 1662)
  - Construiu em 1942 a primeira máquina de calcular, baseada em engrenagens e alavancas, e que permitia fazer adições e subtrações
- Leibniz (1646 - 1716)
  - Construiu outra máquina no mesmo estilo, porém permitia também a realização de multiplicações e divisões

Histórico

Geração Zero – Computadores Mecânicos (1642 - 1945)

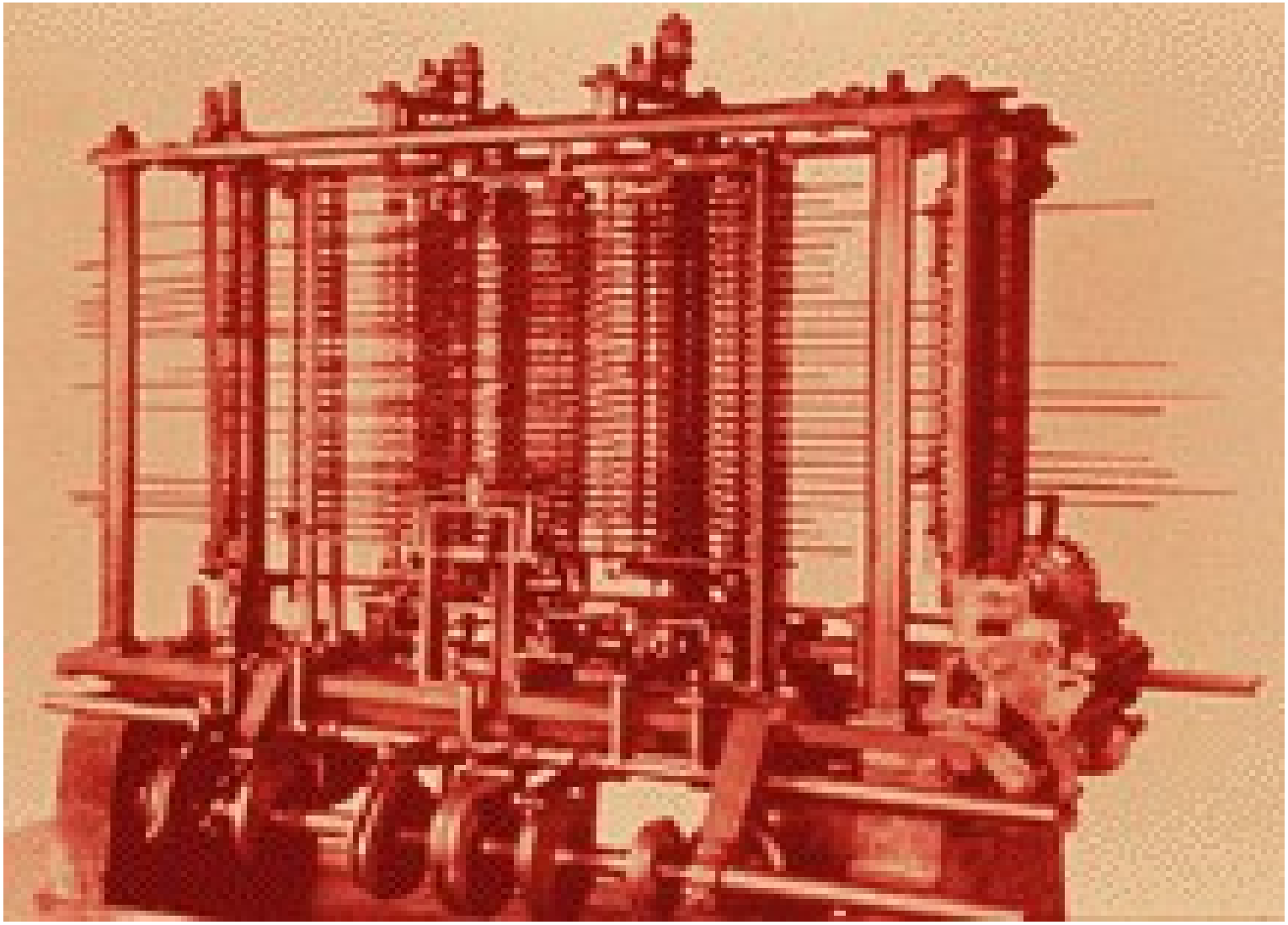
- Charles Babbage (1792 – 1871)
  - Máquina Diferencial: implementava o método de diferenças finitas para navegação naval. A saída era gravada em pratos de aço
  - Máquina Analítica: proposta de uma máquina de propósito geral. Era composta por quatro componentes: memória, unidade de computação, unidade de entrada e unidade de saída

Histórico

Geração Zero – Computadores Mecânicos (1642 - 1945)

- Meados do século XIX: Charles Babbage (1792-1871), por volta de 1833, projetou o primeiro computador digital. No entanto, a pouca tecnologia da época não permitiu que o projeto tivesse sucesso.
  - Máquina analítica:
    - Não tinha um SO;
    - Mas tinha um software que possibilitava seu uso;

Máquina analítica



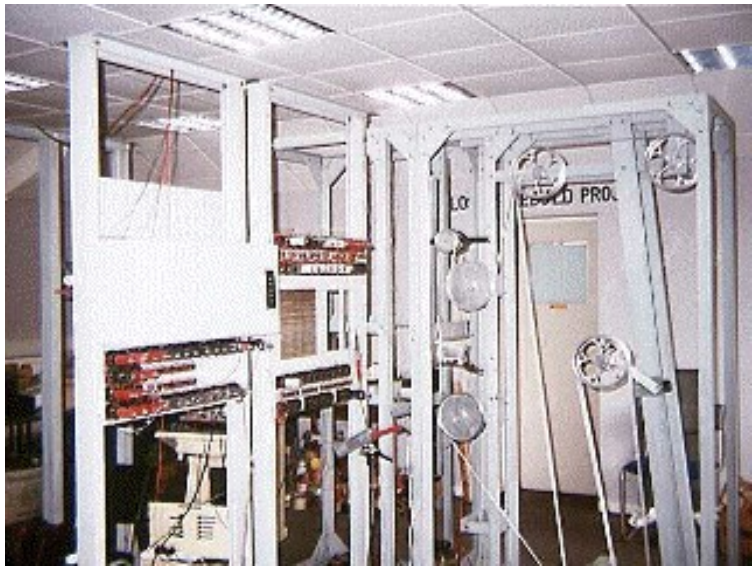
Histórico

Geração Zero – Computadores Mecânicos (1642 - 1945)

- Máquinas a relé
- Konrad Zuse:
  - 1º computador eletromecânico, constituído de relés - efetuava cálculos e exibia os resultados em fita perfurada.
- John Atanasoff e George Stibbitz
  - Construíram no final da década de 1930 calculadoras que já usavam aritmética binária e possuíam memória baseada em capacitores.

Histórico

- **1a. Geração de Computadores (1945 - 1955)**
  - Computadores à Válvula
  - Ausência de um S.O.: a programação era feita diretamente em linguagem de máquina

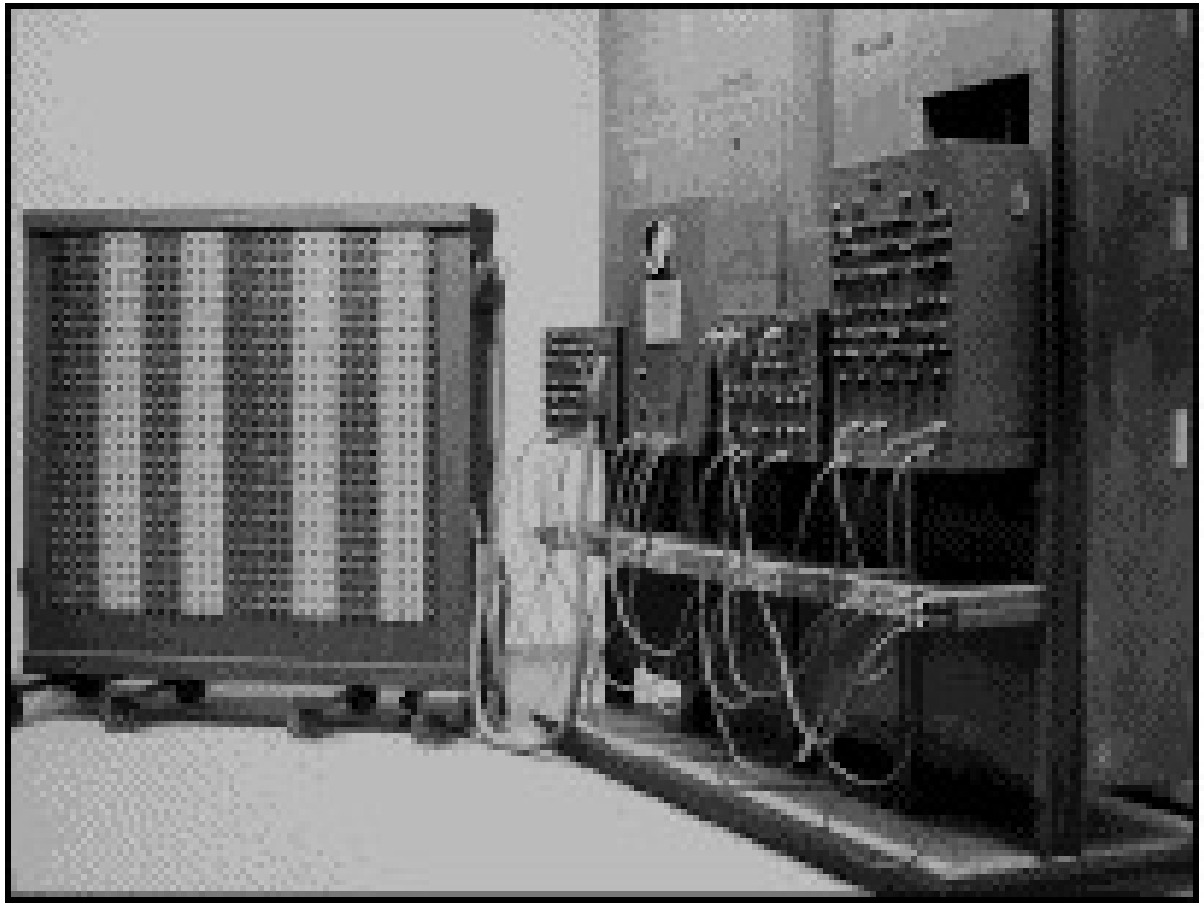


Colossus Mark I

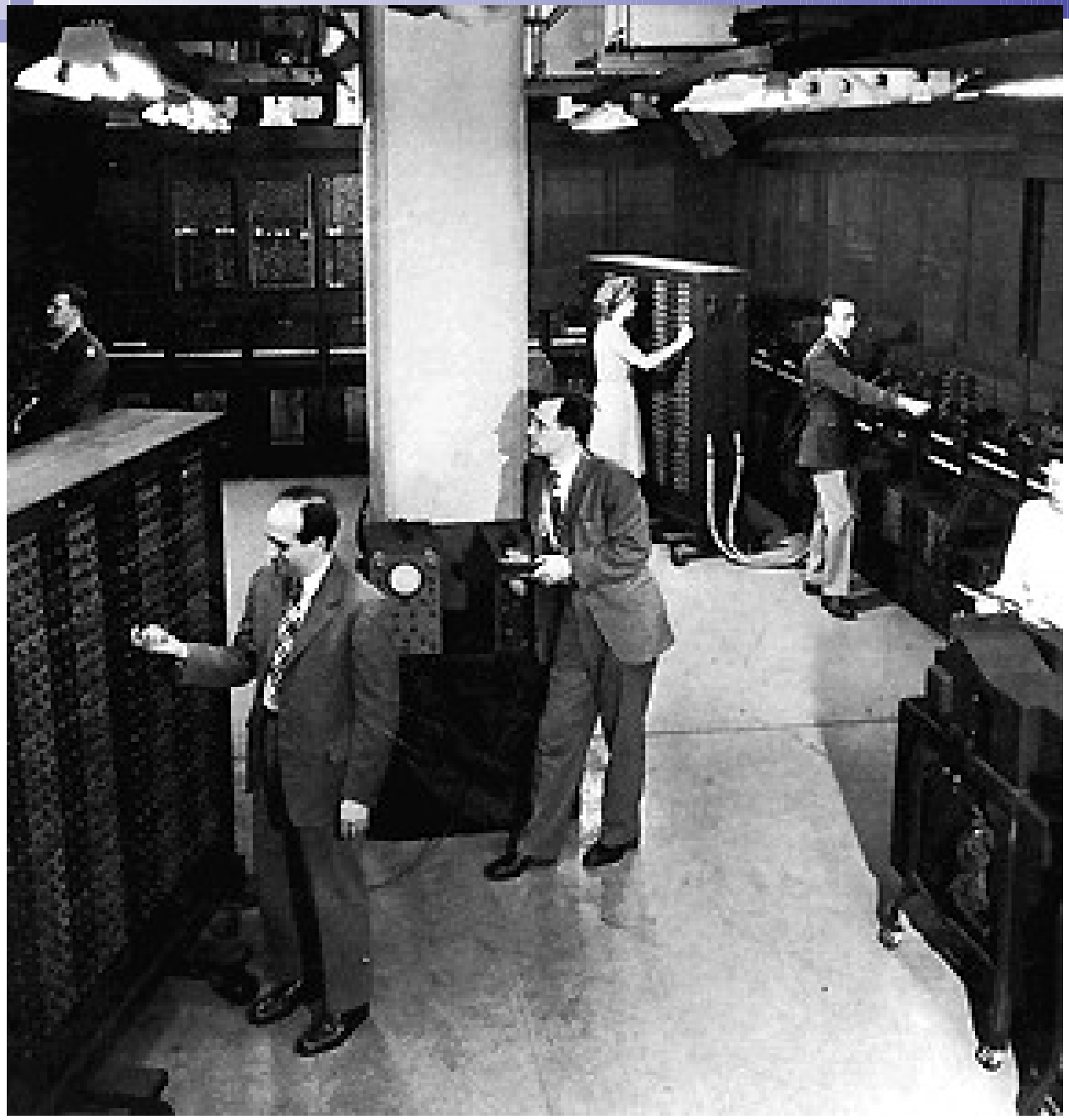
Histórico

- Segunda Guerra Mundial: grande motivador
- COLOSSUS
  - Primeiro computador digital eletrônico construído pelo Governo Britânico em 1943.
  - Objetivo: decodificar as mensagens trocadas pelos alemães durante a Segunda Guerra Mundial, que eram criptografadas por uma máquina chamada ENIGMA.
  - Participação de Alan Turing.
- ENIAC (Electronic Numerical Integrator and Computer)
  - Computador eletrônico construído por John Mauchley e J. Presper Eckert (EUA) em 1946 para fins militares.
  - 18.000 tubos a vácuo; 1.500 relés; 30 toneladas; 140 kilowatts; 20 registradores de números decimais de 10 dígitos
  - Programação feita através de 6.000 switches e de milhares de jumpers (cabos de conexão)
  - Participação de John von Neumann.

Histórico



ENIAC

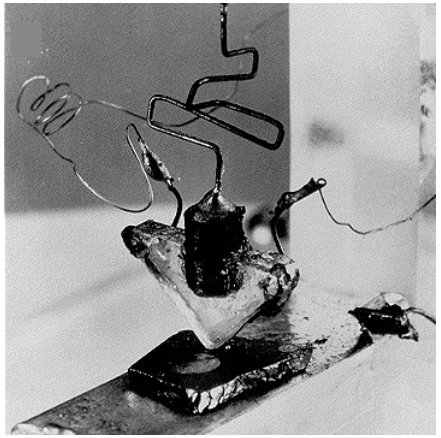


Histórico

- John von Neumann
  - Construiu em 1952 o computador IAS (*Institute for Advanced Study* – Princeton, USA)
  - Programa Armazenado: programas e dados representados de forma digital em memória
  - Processamento baseado em aritmética binária, ao invés de decimal
- Máquina de Von Neumann
  - Componentes: Memória, Unidade Lógica e Aritmética (ULA), Unidade de Controle e os dispositivos de entrada/saída.
  - Memória: 4096 palavras de 40 bits (2 instruções de 20 bits ou um inteiro)
  - Instrução: 8 bits para indicar o tipo, 12 bits para endereçar a memória
  - Acumulador: registrador especial de 40 bits. Tem por função armazenar um operando e/ou um resultado fornecido pela ULA.

Histórico

- **2a. Geração de Computadores (1955 - 1965)**
  - Invenção do Transistor (William Shockley, John Bardeen, e Walter Brattain)



  - Uso da linguagem Assembly e FORTRAN
  - SOs do tipo lote (batch)

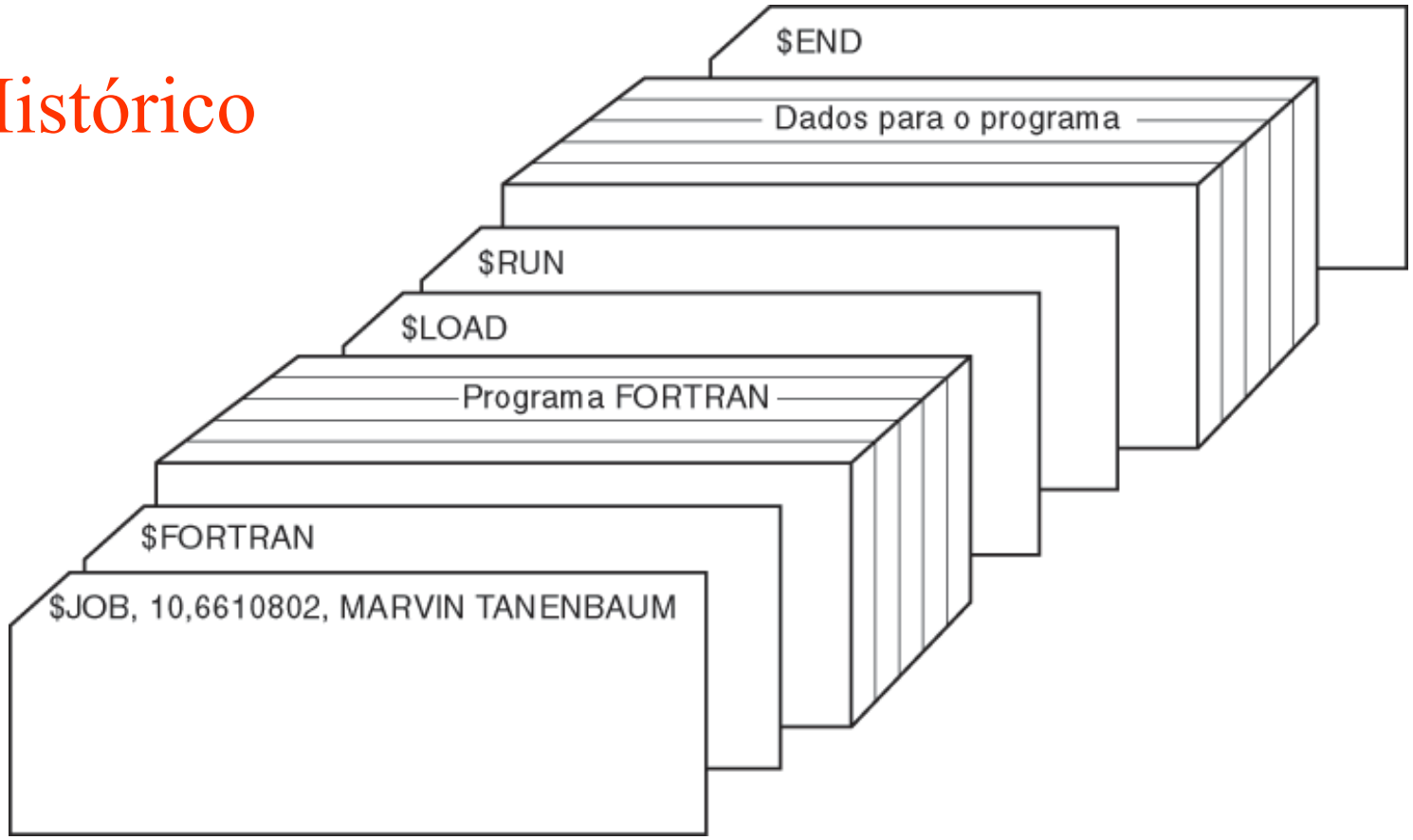
Histórico

- Segunda Geração (1955-1965) – Transistores e Sistemas em *Batch*
  - O desenvolvimento dos transistores tornou o computador mais confiável possibilitando sua comercialização - *Mainframes*;
  - No entanto, devidos aos altos custos poucos tinham acesso a essa tecnologia – somente grandes empresas, órgãos governamentais ou universidades;

Histórico

- Surge a idéia de linguagem de programação de alto nível – Fortran (desenvolvida pela IBM – 1954-1957);
- Cartões perfurados ainda são utilizados
  - Operação: cada programa (*job*) ou conjunto de programas escrito e perfurado por um programador era entregue ao operador da máquina para que o mesmo fosse processado – alto custo
  - Sistemas em *Batch* (lote)
    - Consistia em coletar um conjunto de *jobs* (um ou mais programas) e fazer a gravação desse conjunto para uma fita magnética

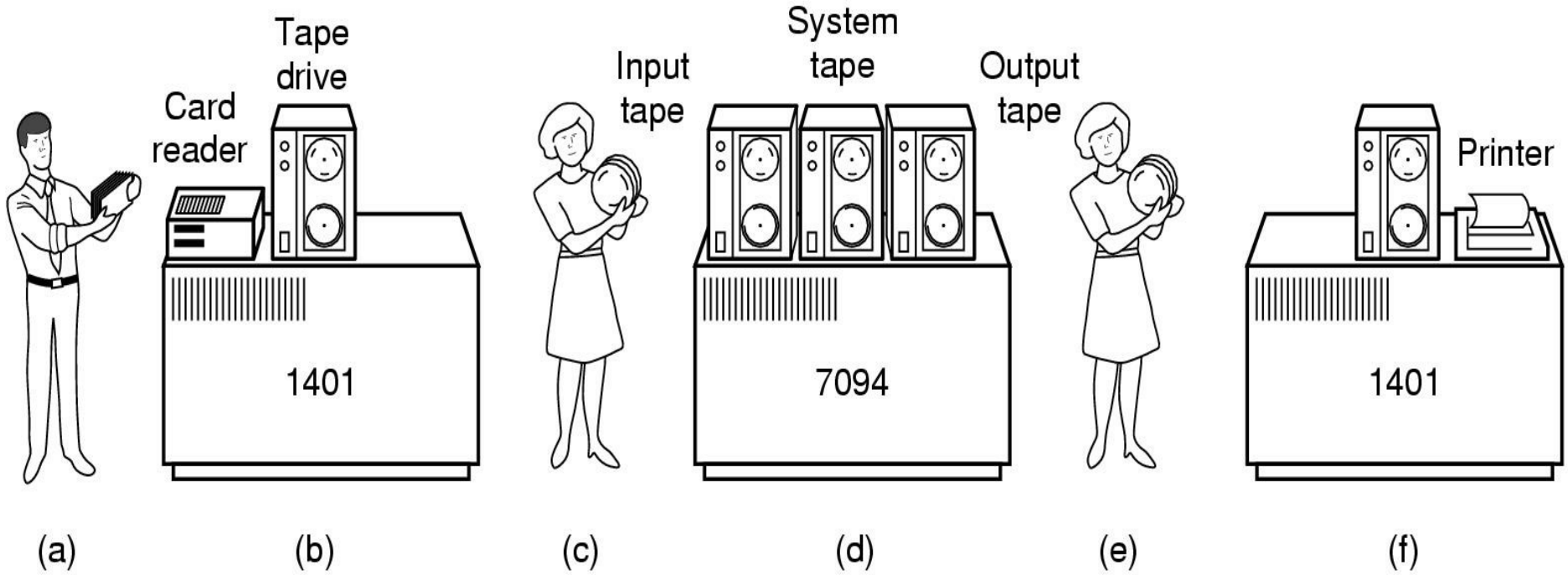
Histórico



Estrutura de um job FMS típico – 2a. geração

Histórico

Sistema em Batch

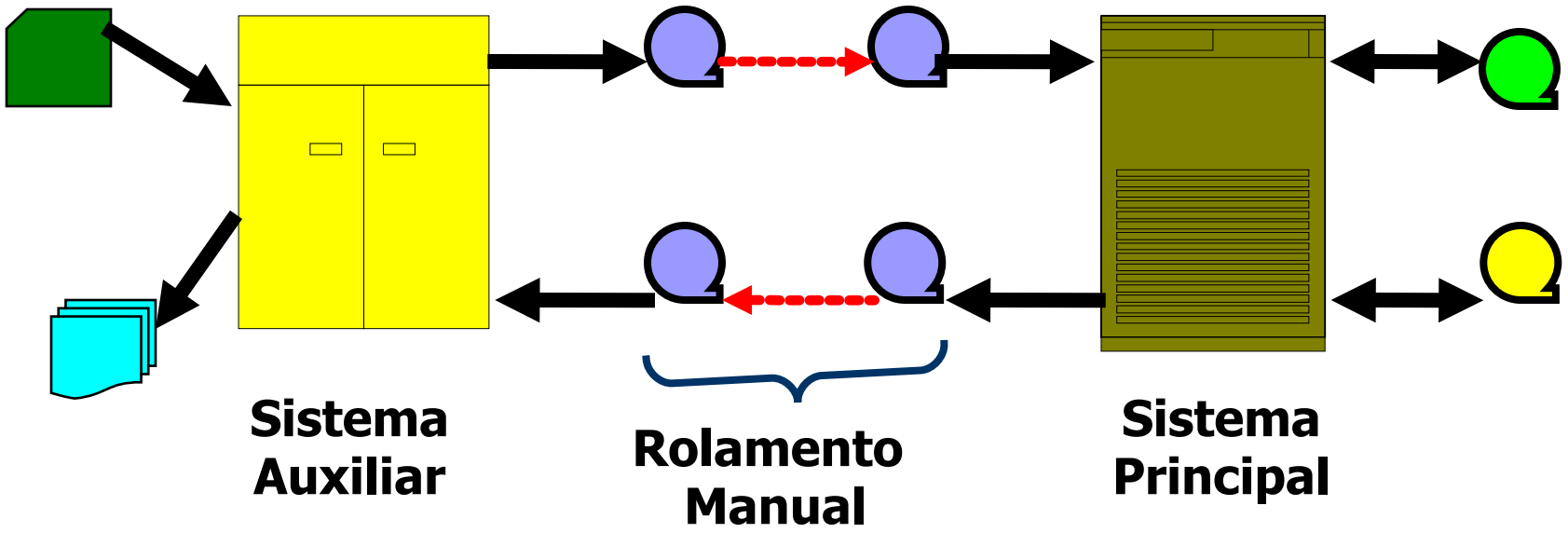


FMS (*Fortran Monitor System*)

Processamento: IBSYS – SO IBM para o 7094

Histórico

- **1957:** uso de sistema auxiliar (técnica do spooling)

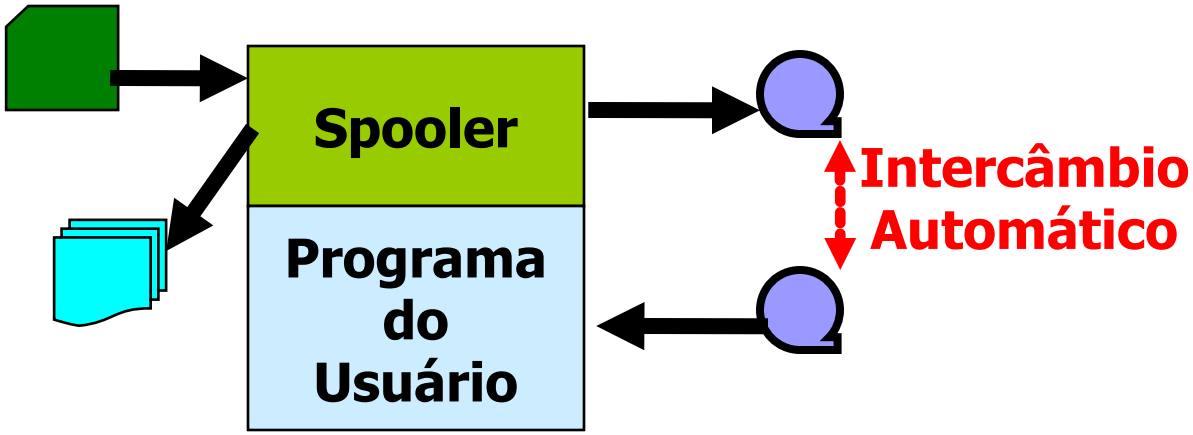


Histórico

- **1959:**
  - Introdução de canais autônomos de Entrada/Saída
  - Criação das Interrupções
  - **Entrada/Saída em paralelo com o cálculo**

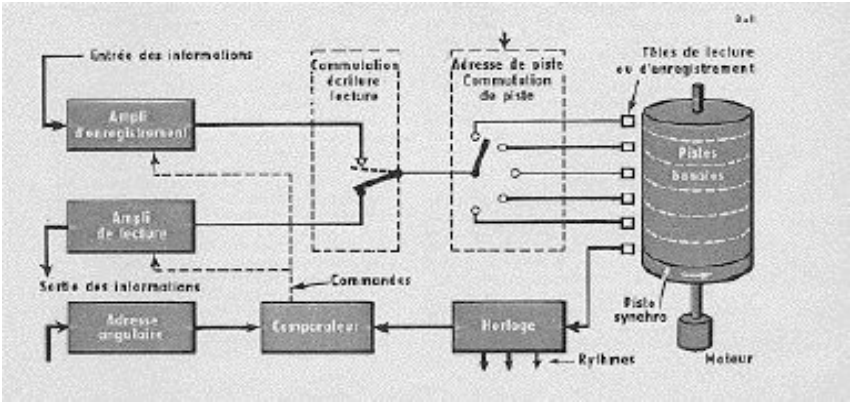
Histórico

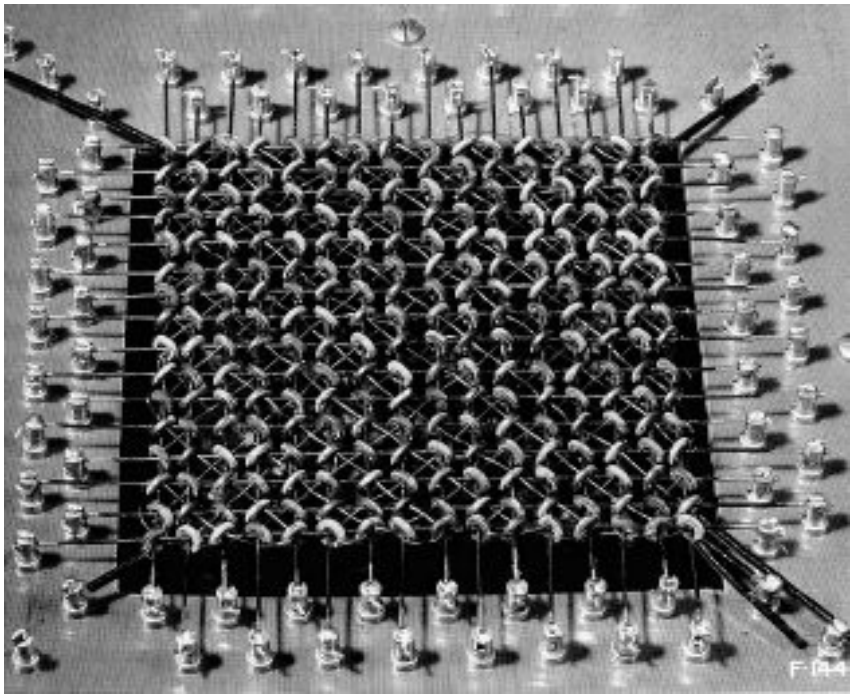
- **1960:**
  - Uso de Spooler automático



  - Invenção dos discos e tambores magnéticos
  - S.O.s Típicos: FMS (Fortran Monitor System) e
  - IBSYS (da IBM)

Exemplos de tecnologia de armazenamento da 2a. geração



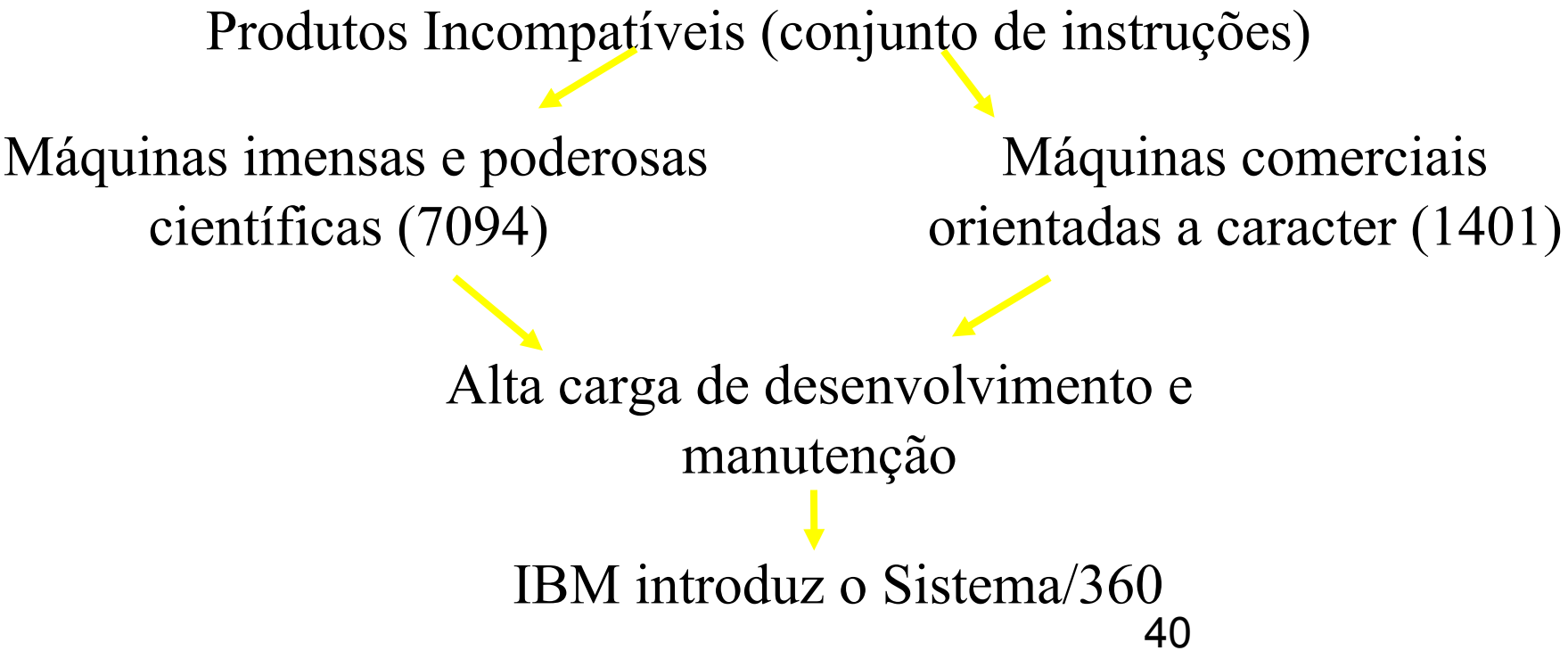


Tambor Magnético

Memória de Ferrite

Histórico (Terceira Geração)

- Terceira Geração (1965-1980) – Circuitos integrados, Multiprogramação e Time-sharing



Histórico

- **Multiprogramação**:
  - Dividir a memória em diversas partes (partições) e alocar a cada uma dessas partes um *job*.
  - Manter na memória simultaneamente uma quantidade de *jobs* suficientes para ocupar 100% do tempo do processador, diminuindo a ociosidade.
  - Importante: o hardware é que protegia cada um dos *jobs* contra acesso indevidos de outros *jobs*.

Histórico

- Mesmo com o surgimento de novas tecnologias, o tempo de processamento ainda era algo crítico. Para corrigir um erro de programação, por exemplo, o programador poderia levar horas

TimeSharing

Histórico

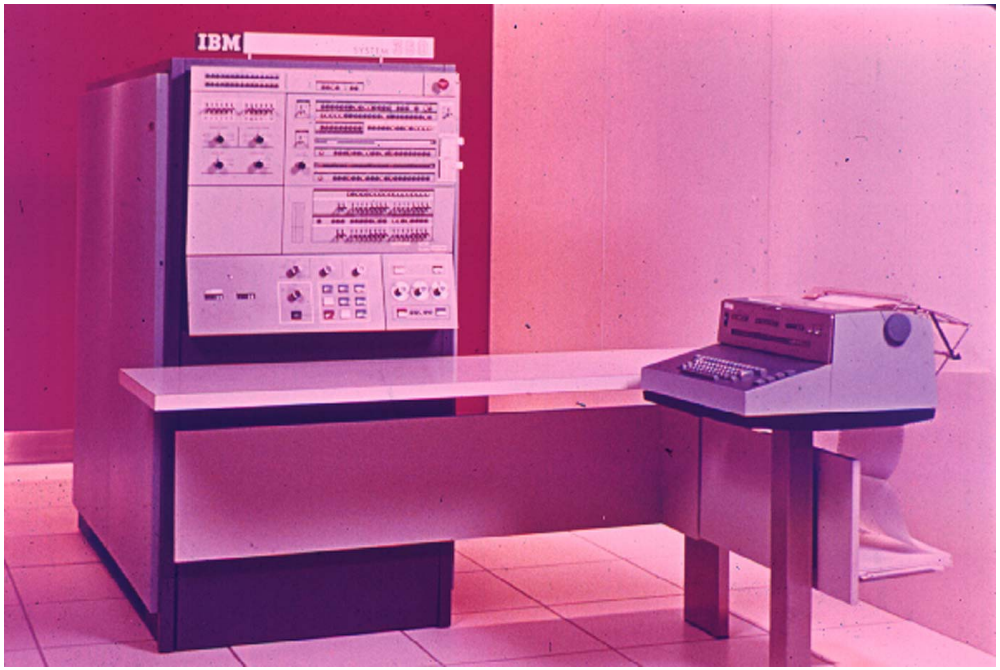
- ***TimeSharing***: cada usuário tinha um terminal *on-line* à disposição;
  - Primeiro sistema *TimeSharing*: CTSS (*Compatible Time Sharing System*) – 7094 modificado.
  - Ex.: se 20 usuários estão ativos e 17 estão ausentes, o processador é alocado a cada um dos 3 *jobs* sendo executados;
- Surge o MULTICS (predecessor do UNIX);
  - POSIX (Portable OS IX) → **Wrapper**
- Família de minicomputadores PDP da DEC;
  - Compatíveis;
  - Unix original rodava no PDP-7 (Ken Thompson – cientista da Bell Labs)

Histórico

- ***Spooling*** (*Simultaneous Peripheral Operation On Line*):
  - Possibilitar que a leitura de cartões de *jobs* fosse feita direta do disco;
  - Assim que um *job* terminava, o sistema operacional já alocava o novo *job* à uma partição livre da memória direto do disco.
  - Impressão.

Histórico

- Invenção dos Circuitos Integrados (chips) com baixa escala de integração (SSI - Small Scale Integration)
- Sistema OS/360 (IBM): 1o. a usar circuitos SSI



Histórico

Sistema GE 625

(SO Multics)

Aula de Hoje

1. **Tipos de Sistemas Operacionais (SOs)**
2. **Estruturas de SOs**

Aula de Hoje (conteúdo detalhado)

1.5 Evolução dos SOs

Quarta e Quinta Geração de Computadores

1.6 Tipos de SOs

1.7 Diferentes Visões de SOs

1.8 Estruturas de SOs

Um Breve Histórico

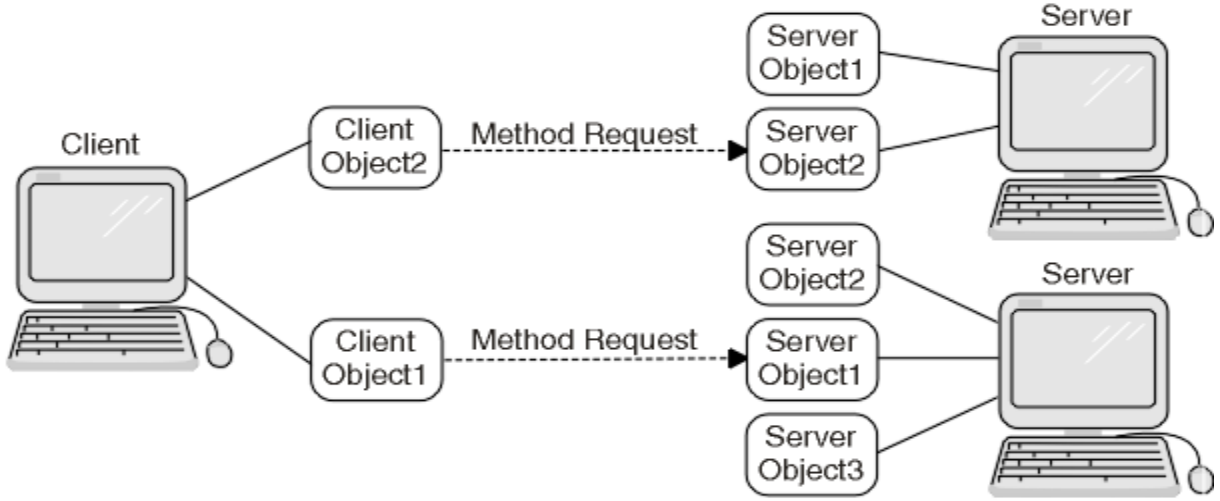
- **4a. Geração de Computadores (1980 - Hoje)**
  - **Invenção dos Circuitos Integrados com alta escala de integração (LSI - Largel Scale Integration)**
  - **Sistemas Operacionais para Microcomputadores**
    - **CP/M (8 bits)**
    - **DOS (16 bits)**
    - **UNIX (32 bits)...**
  - **Sistemas Operacionais de Rede**
  - **Sistemas Operacionais Distribuídos**

Histórico

- 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

Histórico
Quinta Geração (1990-hoje)

- 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



Histórico
Quinta Geração

- O protocolo de comunicações TCP/IP tornou-se largamente utilizado (Depto de Defesa dos EUA) e as LANs (*Local Area Networks*) tornaram-se mais práticas e econômicas com o surgimento do padrão ***Ethernet*** desenvolvido pela Xerox;
- Desenvolvimento e popularização do modelo cliente/servidor;
- Difusão das redes de computadores
  - ***Internet***

Histórico
Quinta Geração

- Sistemas Operacionais Distribuídos:
  - Apresenta-se como um sistema operacional centralizado, mas que, na realidade, tem suas funções executadas por um conjunto de máquinas independentes; cria uma "ilusão" ao usuário
- Descentralização do controle;
- Linux;
- Família Windows (Vista, 7, 8, 10);
- Sistemas Operacionais em Rede – não são diferentes dos SOs para os monoprocessadores.

Atualidades

- Sistemas Operacionais Orientados a Objetos
  - Reúso
  - Interface orientada a objetos
- JavaOS
  - Portabilidade;
- Sistemas Operacionais de Tempo Real
  - Importante:
    - Gerenciamento de Tempo (críticos e não críticos);
    - Gerenciamento de processos críticos (aviões, caldeiras);
  - RTLinux (Real Time Linux);
    - http://www.fsmlabs.com/
- Sistemas Operacionais Embarcados: telefones, aparelhos eletrodomésticos; PDAs;

Aula de Hoje (conteúdo detalhado)

1.5 Evolução dos SOs

Quarta e Quinta Geração de Computadores

1.6 Tipos de SOs

1.7 Diferentes Visões de SOs

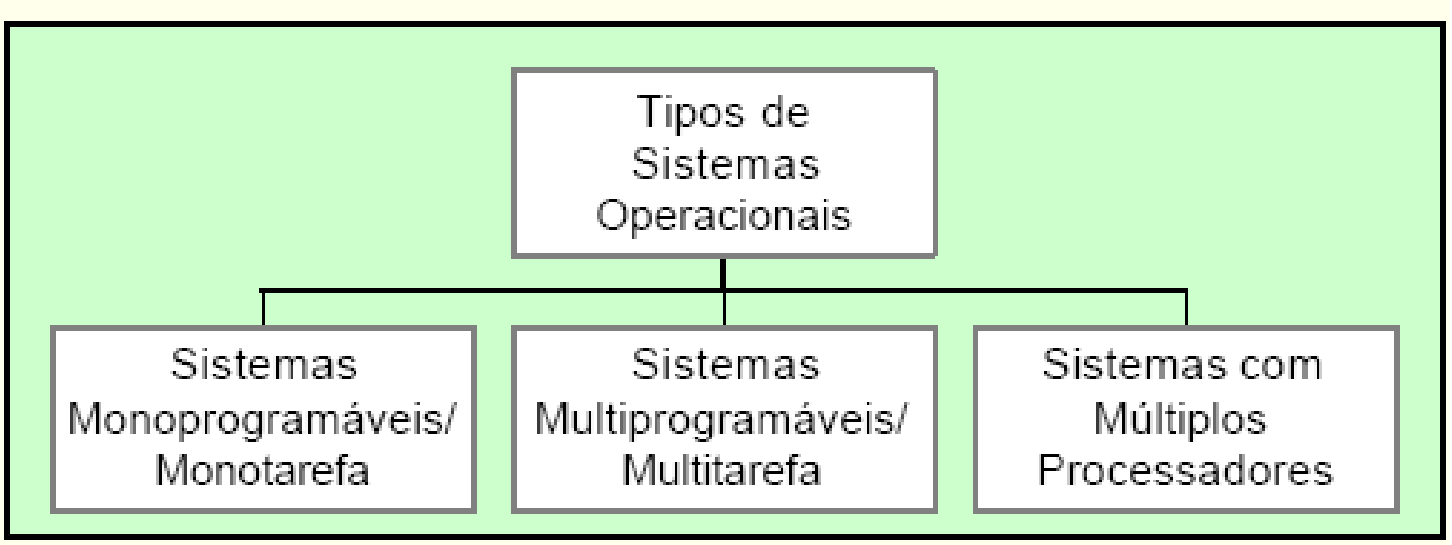
1.8 Estruturas de SOs

Introdução

1.6 Tipos de Sistemas Operacionais

- **Classificação quanto ao compartilhamento de hardware**
  - **Sistemas Operacionais Monoprogramados**
    - **Só permite um programa ativo em um dado período de tempo, o qual permanece na memória até seu término**
    - **Ex: DOS**
  - **Sistemas Operacionais Multiprogramados**
    - **Mantém mais de um programa simultaneamente na memória principal, para permitir o compartilhamento efetivo do tempo de UCP e demais recursos**
    - **EX: Unix, VMS, Windows, etc.**

Introdução



- **SOs Monoprogramáveis ou Monotarefa**
  - *Se caracterizam por permitir que o processador, a memória e os periféricos permaneçam exclusivamente dedicados à execução de um único programa. Recursos são mal utilizados, entretanto é fácil de ser implementado.*

Introdução

- **SOs Multiprogramáveis ou Multitarefa**
  - Nestes SOs vários programas dividem os recursos do sistema. As vantagens do uso destes sistemas são o aumento da produtividade dos seus usuários e a redução de custos, a partir do compartilhamento dos diversos recursos do sistema.
  - Podem ser **Multiusuário** (mainframes, mini e microcomputadores) ou **Monousuário** (PCs e estações de trabalho). É possível que ele execute diversas tarefas concorrentemente ou mesmo simultaneamente (**Multiprocessamento**) o que caracterizou o surgimento dos SOs **Multitarefa**.

Introdução

- Os SOs **Multiprogramáveis/Multitarefa** podem ser classificados pela forma com que suas aplicações são gerenciadas, podendo ser divididos conforme mostra o gráfico.

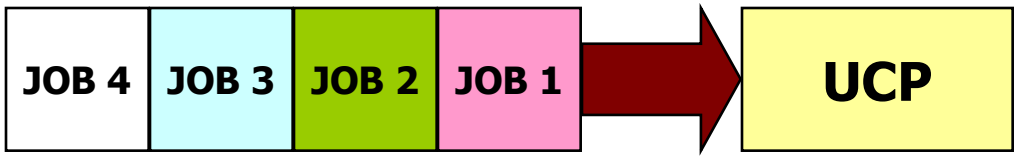
Introdução

Classificação quanto a interação permitida

- fator determinante → Tempo de resposta

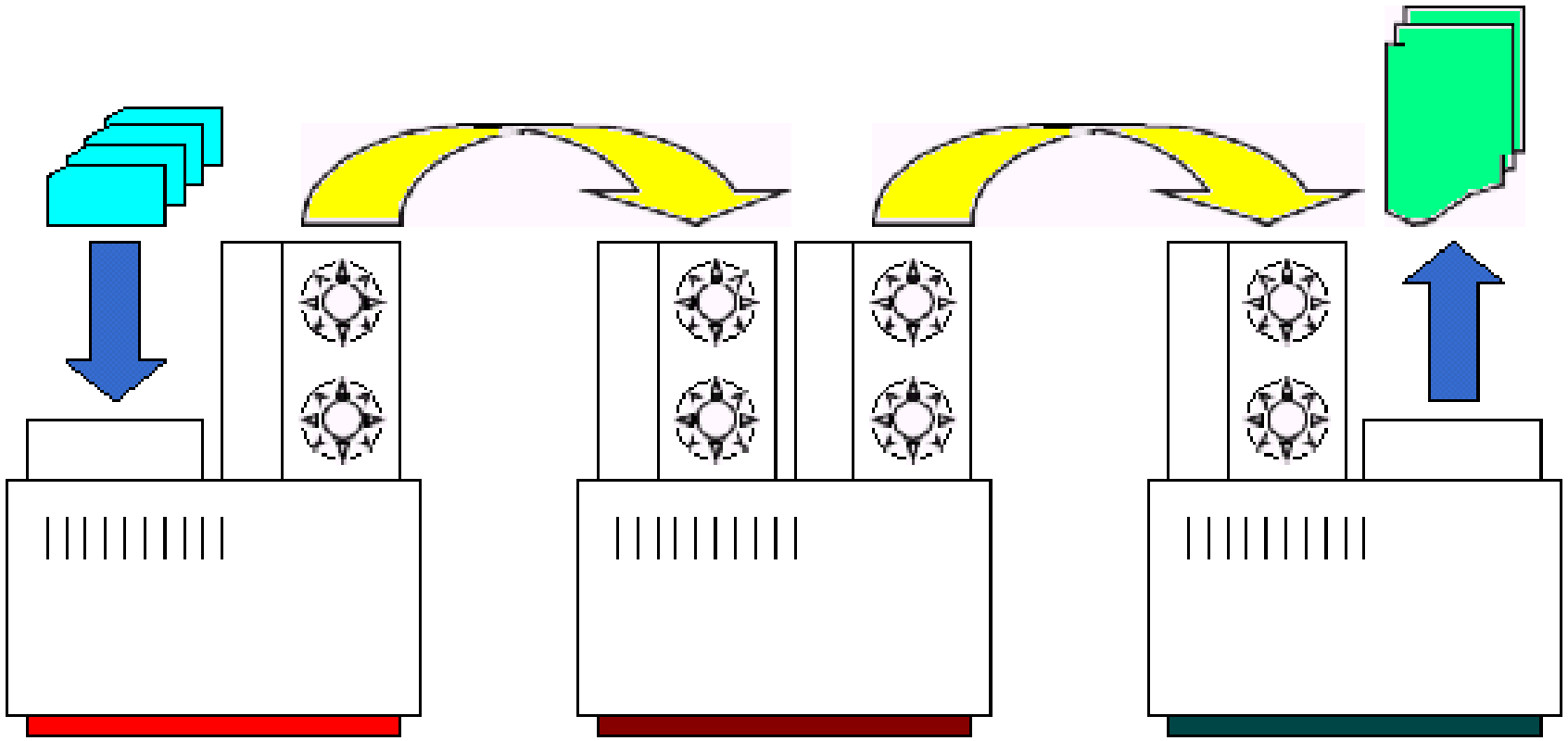
S.O. para processamento em Batch (lote)

- **Os jobs dos usuários são submetidos em ordem sequencial para a execução**
- **Não existe interação entre o usuário e o job durante sua execução**



Introdução

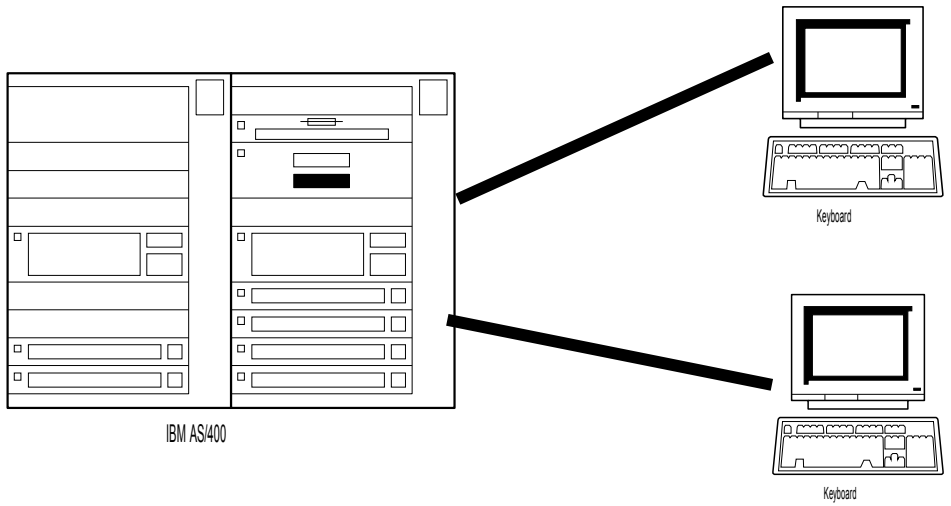
- **S.O. para processamento em Batch (lote)**



Sistema Batch (processamento em lote)

Introdução

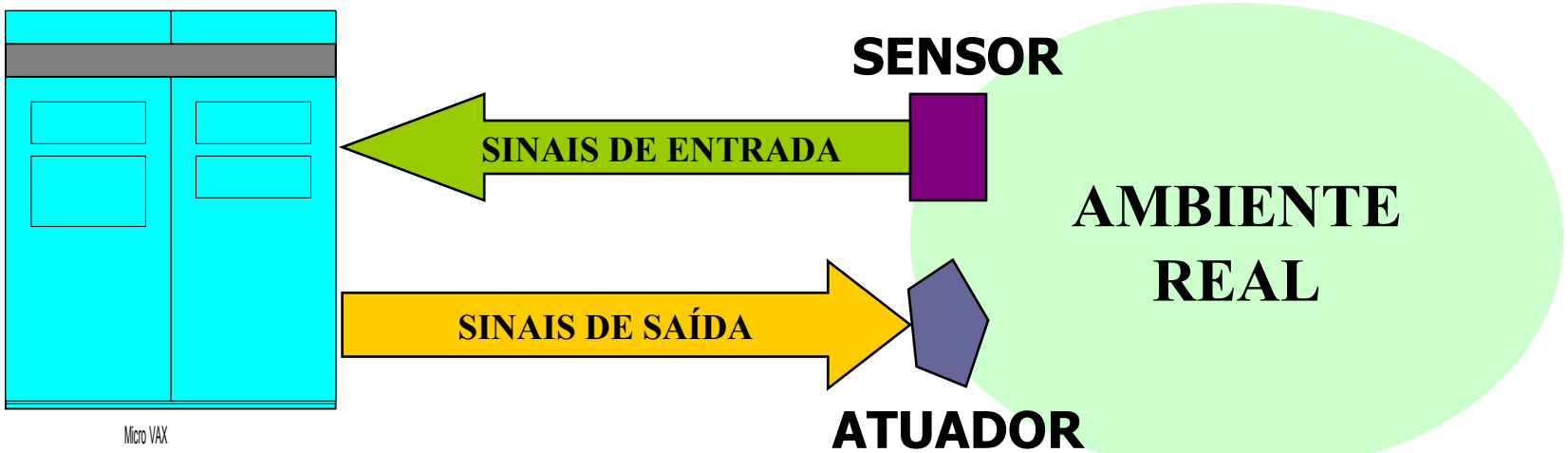
- **S.O. Interativo**
  - **O sistema permite que os usuários interajam com suas computações na forma de diálogo**
  - **Podem ser projetados como sistemas mono-usuários ou multi-usuários (usando conceitos de multiprogramação e time-sharing)**



Terminais

Introdução

- **S.O. de Tempo Real**
  - **Usados para servir aplicações que atendem processos externos, e que possuem tempos de resposta limitados**
  - **Geralmente sinais de interrupções comandam a atenção do sistema**
  - **Geralmente são projetados para uma aplicação específica**



Introdução

- **Classificação segundo o Porte (Tanenbaum, 2003)**
  - **S.O.s de Computadores de grande porte**
  - **S.O.s de Servidores**
  - **S.O.s de Multiprocessadores**
  - **S.O.s de Computadores Pessoais**
  - **S.O.s de Tempo Real**
  - **S.O.s embarcados**
  - **S.O.s de cartões inteligentes**

Aula de Hoje (conteúdo detalhado)

1.5 Evolução dos SOs

Quarta e Quinta Geração de Computadores

1.6 Tipos de SOs

1.7 Diferentes Visões de SOs

1.8 Estruturas de SOs

Introdução

1.7 Diferentes Visões de um S.O.

- **Visão do Usuário da Linguagem de Comando**
  - **As linguagens de comando são específicas de cada sistema**

Classe Funcional	Operações Típicas
Ativação de Programa e Controle	Carregar (Load) Executar (Run) Abortar (abort) Destruir processo (kill)
Gerência de Arquivos	Copiar (Copy, cp,...) Renomear (Ren) Listar diretório (Dir, ls,...)
...	...

Introdução

1.7 Diferentes Visões de um S.O.

- **Visão do Usuário das Chamadas do Sistema**
  - **Permitem um controle mais eficiente sobre as operações do sistema e um acesso mais direto sobre as operações de hardware (especialmente a E/S).**

Tipos Principais de Chamadas
Iniciação de dispositivos
Execução e controle de programas
Serviços de alocação e reserva de recursos do sistema (ex: memória)
Comunicação com dispositivos de E/S
etc.

Aula de Hoje

1. **Estruturas de Sos**
2. **Componentes Básicos de um Sistema**
3. **Processos (Conceitos Básicos)**

Aula de Hoje (conteúdo detalhado)

1 Estruturas de Sos
2. Componentes Básicos (CPU, memória, ..)
3. BIOS
4. Arquitetura do Sistema
5. Processos (Conceitos Básicos)

Introdução

1.8 Estrutura de Sistemas Operacionais

- **Como os sistemas operacionais são normalmente grandes e complexas coleções de rotinas de software, os projetistas devem dar grande ênfase a sua organização interna e estrutura**

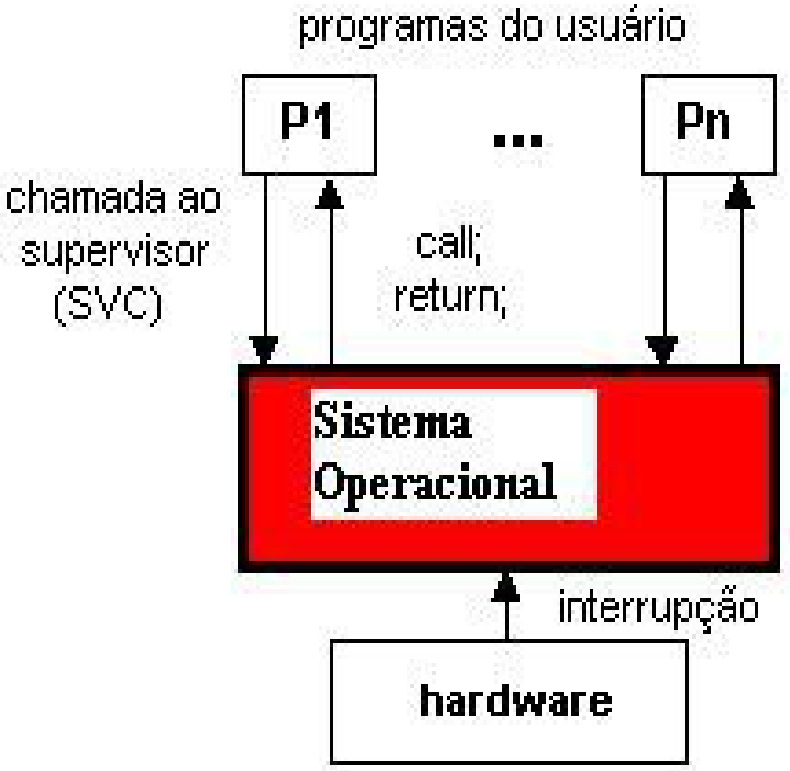
Modo Kernel e Modo Usuário

- Um programa executando em Modo Kernel tem acesso total ao hardware sem o uso do SO
  - Acesso ao CPU e a qualquer endereço da memória RAM
  - Ex.: drivers de impressora
  - O sistema torna-se suscetível a erros (e.g. sistema “travado”)
- Um programa executando em Modo Usuário tem acesso ao hardware por meio do SO
  - A maior parte das aplicações faz uso deste modo

Introdução

1.8.1 Estrutura Monolítica

- **É a forma mais primitiva de S.O.**
- **Consiste de um conjunto de programas que executam sobre o hardware, como se fosse um único programa.**
- **Os programas de usuário podem ser vistos como sub-rotinas, invocadas pelo S.O., quando este não está executando nenhuma das funções do sistema**
- **Ex.: FreeBSD, Linux, Windows Solaris**

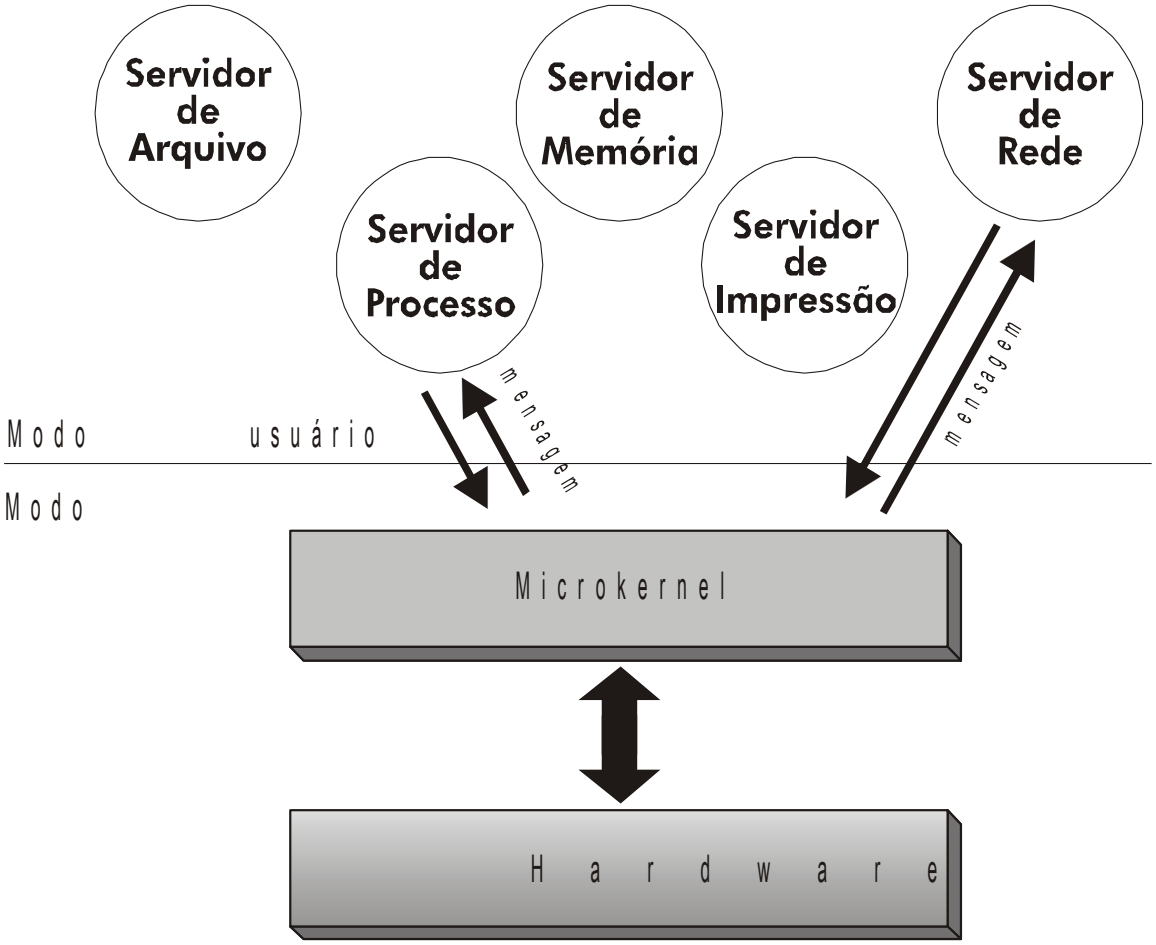


Introdução

1.8.2 Estrutura do MicroKernel

- **MicroNúcleo (microkernel): incorpora somente as funções de baixo nível mais vitais**
- **O microkernel fornece uma base sobre a qual é construído o resto do S.O.**
- **A maioria destes sistemas são construídos como coleções de processos concorrentes**
- **Fornece serviços de alocação de UCP e de comunicação aos processos (IPC).**
- **Ex.: MINIX 3 com 6K LoC**

Estrutura do MicroKernel



Introdução

1.8.3 Sistemas de Camadas - Estrutura Hierárquica de Níveis de Abstração

Os princípios utilizados nesta abordagem são:

- **Modularização:** **divisão de um programa complexo em módulos de menor complexidade. Os módulos interagem através de interfaces bem definidas.**
- **Conceito de "Informação Escondida":** **os detalhes das estruturas de dados e algoritmos são confinados em módulos. Externamente, um módulo é conhecido por executar uma função específica sobre objetos de determinado tipo.**

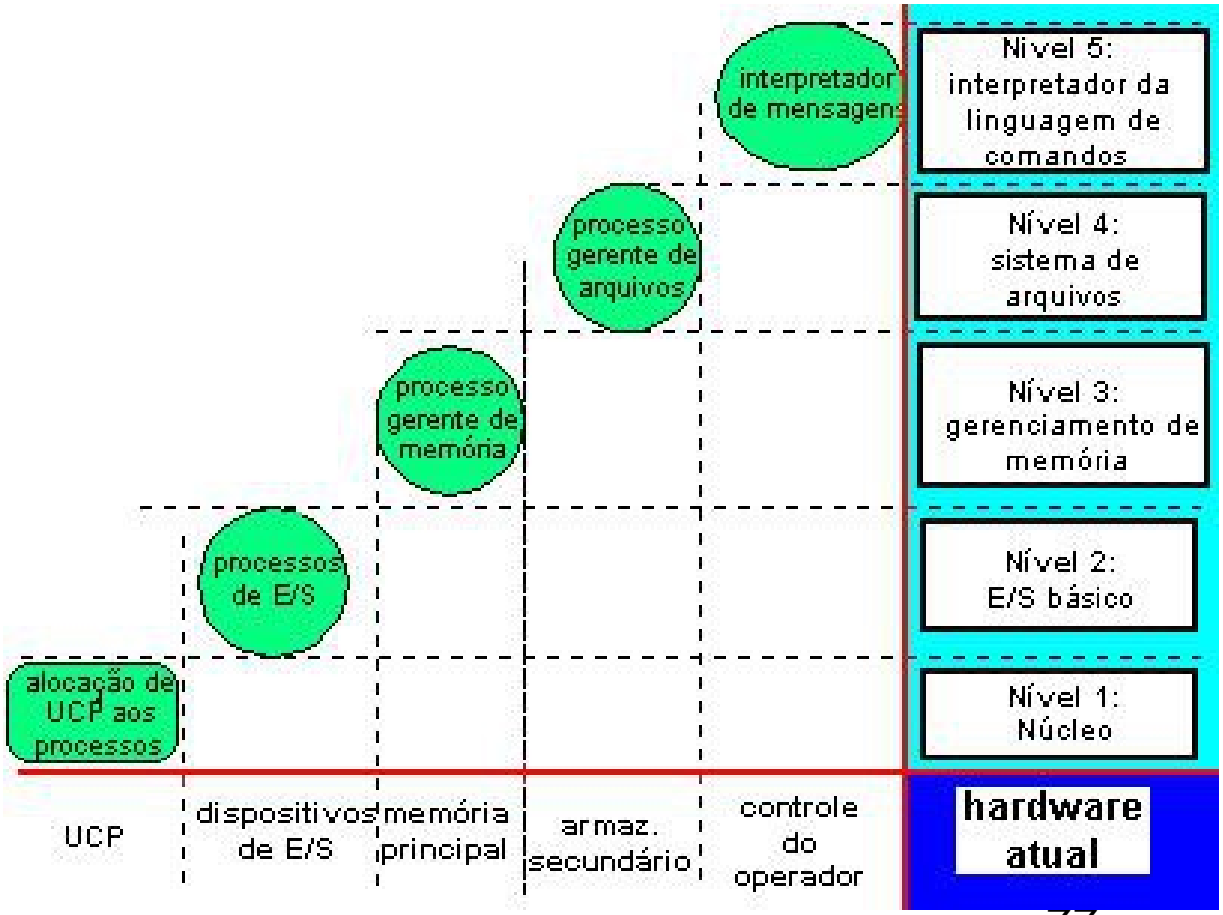
Introdução

Estrutura Hierárquica de Níveis de Abstração

- **A idéia básica é criar um S.O. como uma hierarquia de níveis de abstração, de modo que, a cada nível, os detalhes de operação dos níveis inferiores possam ser ignorados. Através disso, cada nível pode confiar nos objetos e operações fornecidas pelos níveis inferiores.**
- **Importante: interface única**

Introdução

Estrutura Hierárquica de Níveis de Abstração

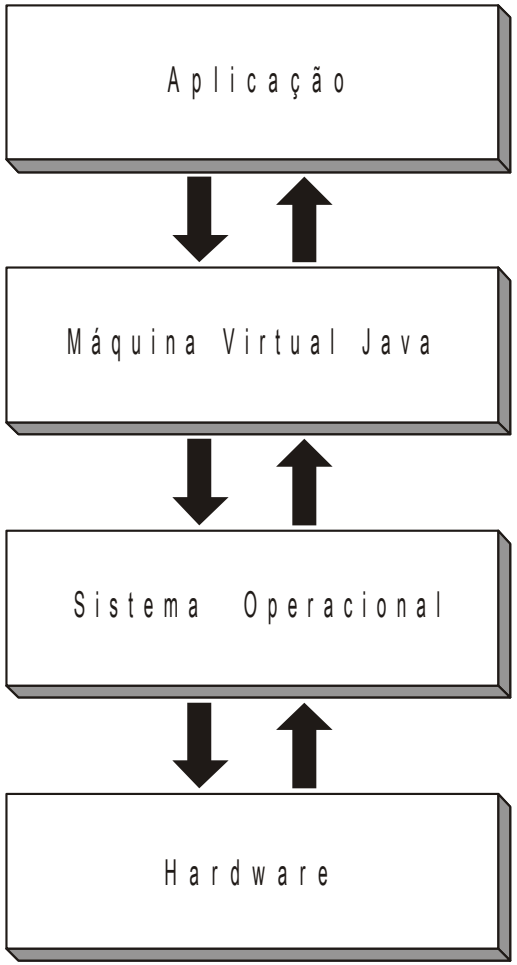


Introdução

1.8.4 Máquina virtual

- O Modelo de Máquina Virtual ou *Virtual Machine* (VM), cria um nível intermediário entre o hardware e o S.O., denominado Gerência de Máquinas Virtuais.
- Este nível cria diversas máquinas virtuais independentes, onde cada uma oferece uma cópia virtual do *hardware*, incluindo modos de acesso, interrupções, dispositivos de E/S, etc.
- Como cada VM é independente das demais, é possível que tenha seu próprio S.O.

Um outro exemplo de utilização desta estrutura ocorre na linguagem Java. Para executar um programa Java é necessário uma máquina virtual Java (*Java Virtual Machine* – JVM)



Perguntas?