Sistemas Operacionais

Parte 1

Um pouco de Historia

Fatos e Geraram tendencias

Prof. Marcos Monteiro

Conceitos básicos

- Computador é...

Conceitos básicos

- Computador é...

Maquina que processa dados?

Que tem:

Hardware

Software

Processamento de DADOS

2 + 3 = 5

Dado (comando) Dado = Informação

:) Marcos Monteiro

Conceitos básicos

COMANDO = SOFTWARE

Código Fonte

```
public class HelloWorld
{
    public static void main( String[] args )
        {
            System.out.println( “Olá  Mundo !!!!!!" );
        }
}
```

```
Olá  Mundo !!!!!!
```

Conceitos básicos

- Computador é...

Maquina **PROGRAMAVEL** que processa dados?

Que tem:

Hardware

Software

Soroban ou Ábaco

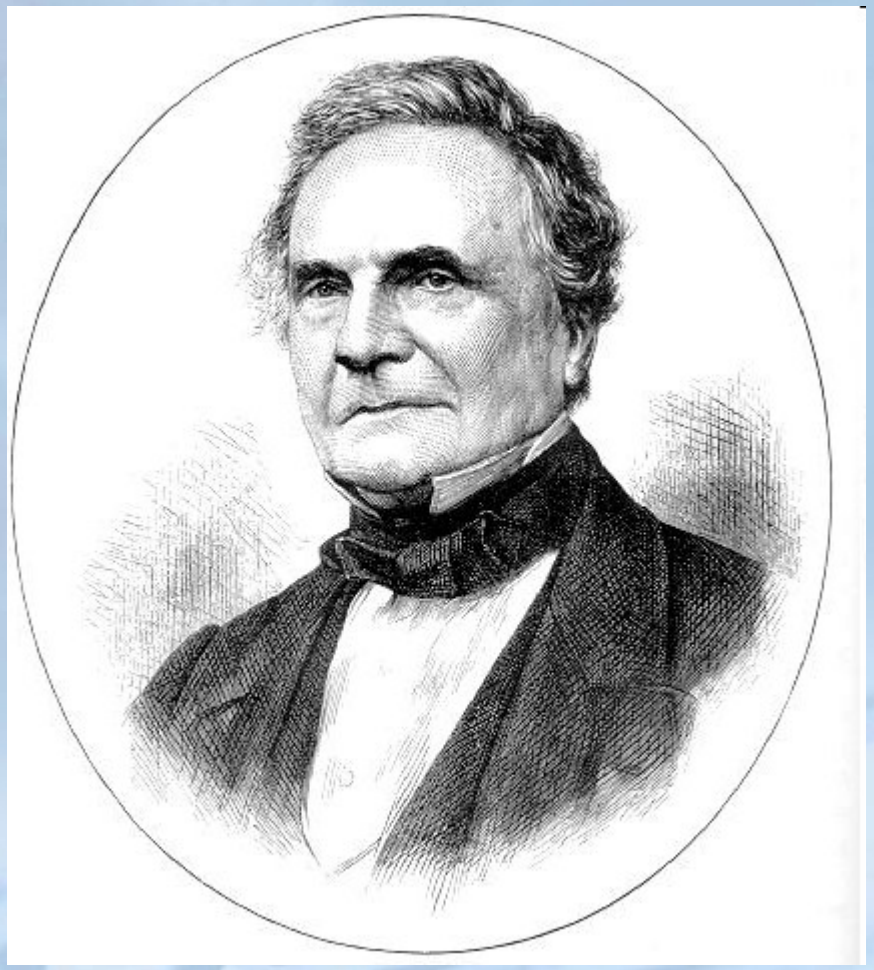
- Teve origem provavelmente na Mesopotâmia, há mais de 5.500 anos.

:) Marcos Monteiro

1642 - Blaise Pascal

:) Marcos Monteiro

1822 - Charles Babbage



Augusta Ada King,
Condessa de Lovelace

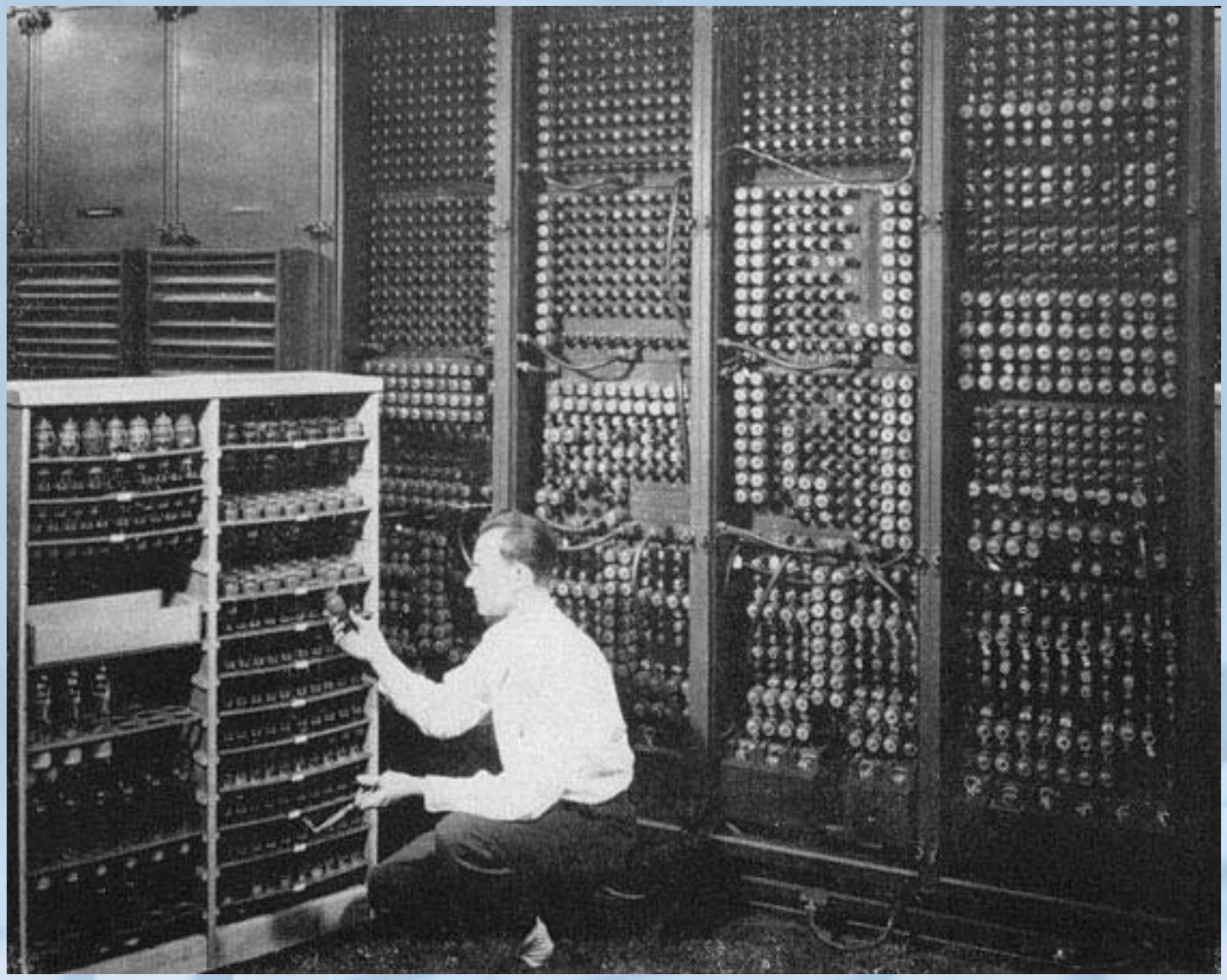
:) Marcos Monteiro

1889 - Hermann Hollerith

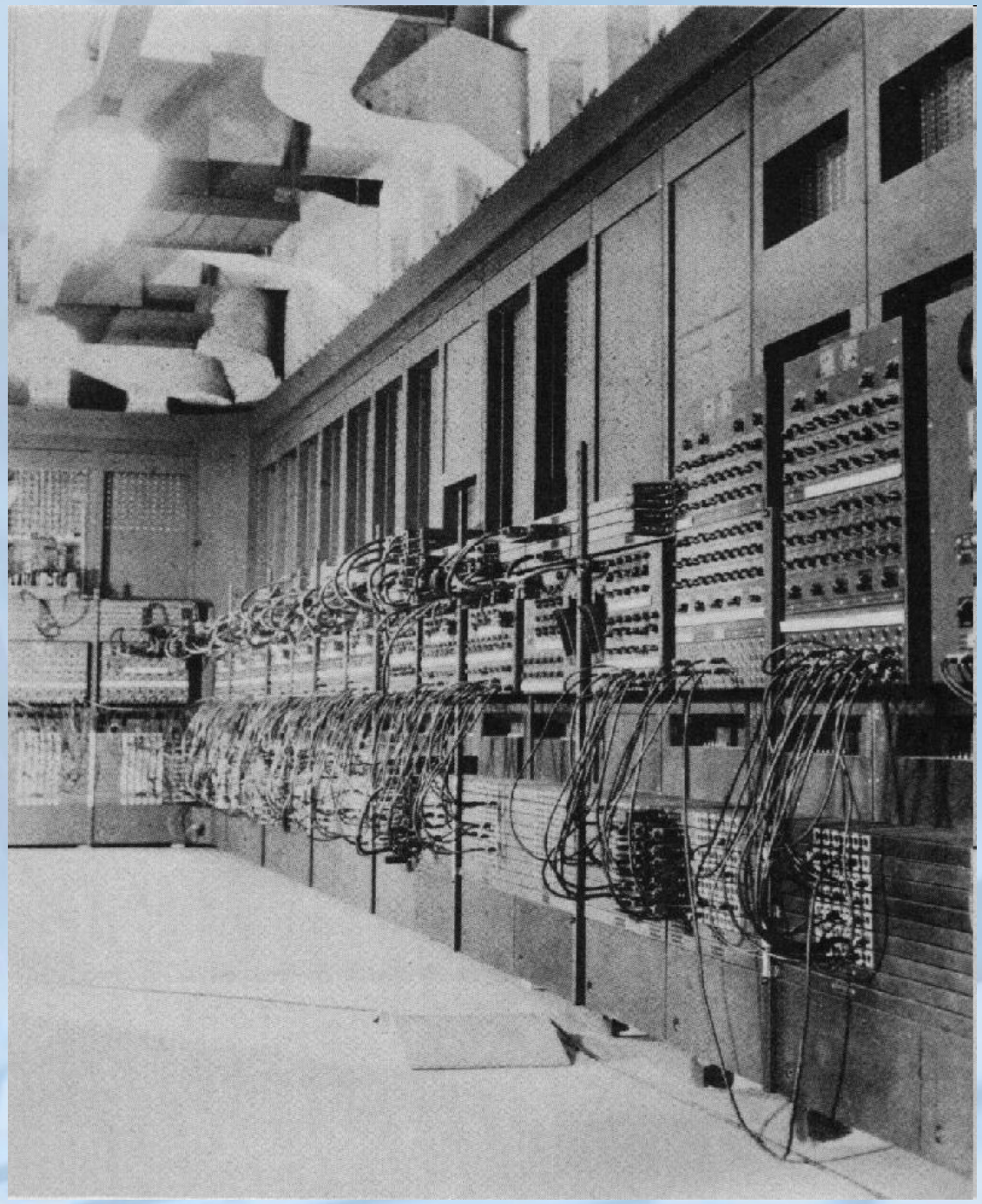
:) Marcos Monteiro

:) Marcos Monteiro

1946 - ENIAC



:) Marcos Monteiro



:) Marcos Monteiro

Os mainframes surgiam cada vez maiores e caros, sendo utilizados apenas por grandes empresas.

1965 – Multics, Unics, digo UNIX

Ken Thompson e Dennis Ritchie

:) Marcos Monteiro

PC - Personal Computer

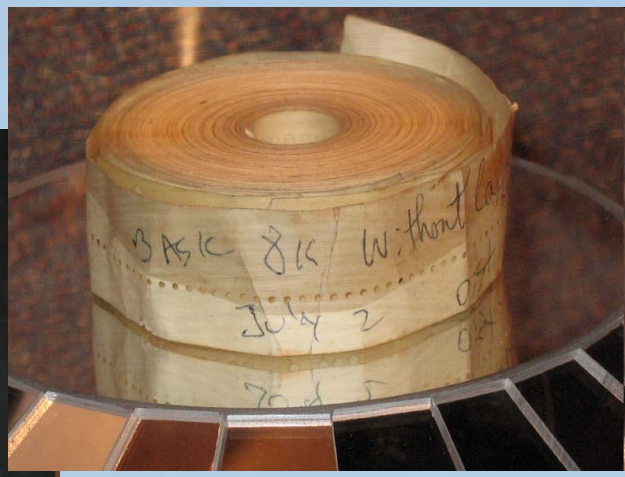
- Segundo o Computer History Museum, o primeiro "computador pessoal" foi o **Kenbak-1**, lançado em 1971. Tinha 256 bytes de memória e foi anunciado na revista Scientific American por US$ 750; todavia, não possuía CPU e era, como outros sistemas desta época, projetado para uso educativo (ou seja, demonstrar como um "computador de verdade" funcionava).

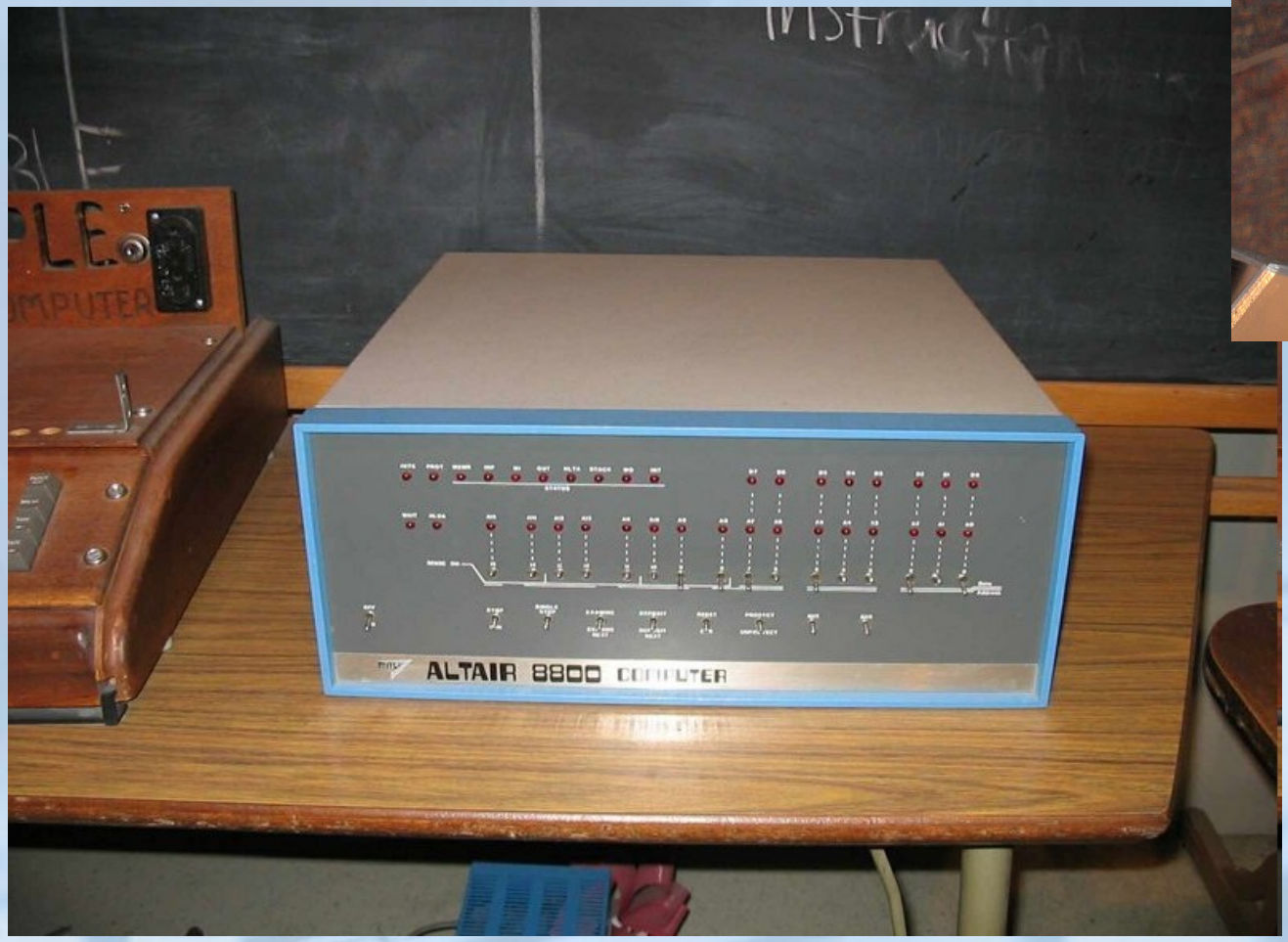
1971 - Kenbak-1

1975 - Altair 8800

MITS Altair 8800 é um computador pessoal projetado em 1975, baseado na CPU Intel 8080.

Altair BASIC

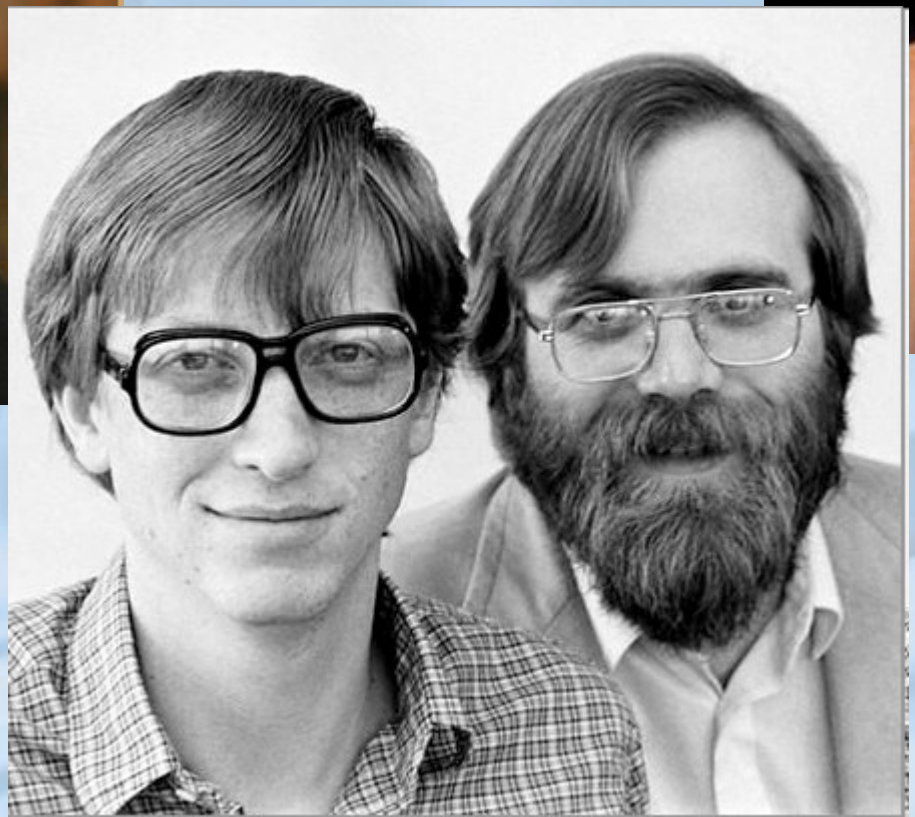




:) Marcos Monteiro

1975 -MicroSoft

Bill Gates e Paull Allen

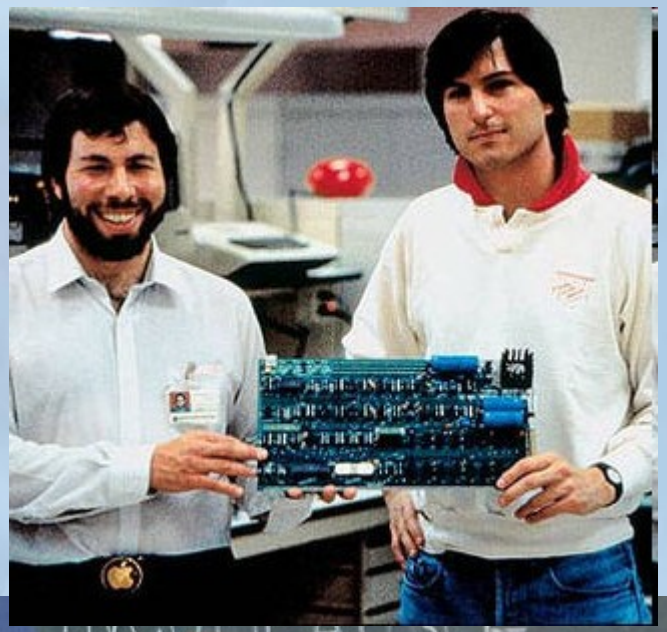


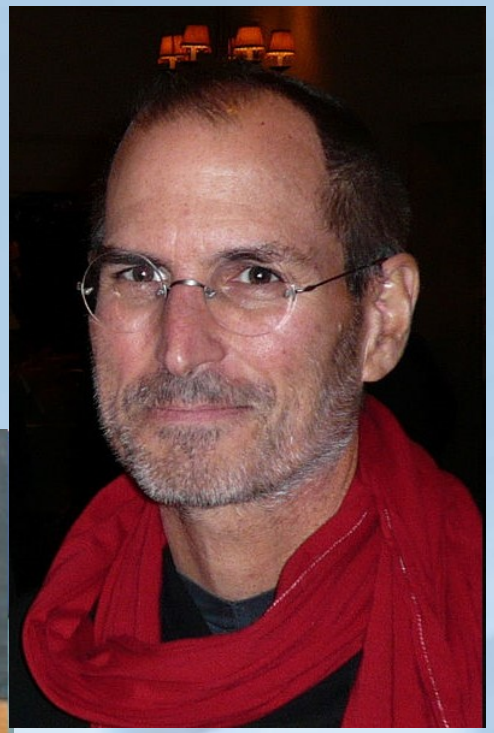
:) Marcos Monteiro


ALBUQUERQUE N MEX
APD 105 519
12 13 77

1976 - Apple

Steve Wozniak e Steve Jobs





:) Marcos Monteiro

O Apple II foi lançado em 1977 com teclado integrado, gráficos coloridos, sons, gabinete de plástico e oito slots de expansão.

:) Marcos Monteiro

1981 - IBM-PC

:) Marcos Monteiro

1984 - Macintosh

:) Marcos Monteiro

iMac

:) Marcos Monteiro

O que é Software Livre?

- **Richard Stallman**
- **FSF (1985) http://www.fsf.org/**

FREE SOFTWARE FOUNDATION

- **Projeto Gnu**

- **GPL (General Public License)**

Em termos gerais, a GPL (**General Public License**) baseia-se em 4 liberdades:

- A liberdade de **executar** o programa, para qualquer propósito;
- A liberdade de **estudar** como o programa funciona e adaptá-lo para as suas necessidades.
- A liberdade de **redistribuir** cópias.
- A liberdade de **aperfeiçoar** o programa, e liberar os seus aperfeiçoamentos.

•CopyLeft

:) Marcos Monteiro

1987 - Minix

Andrew Stuart Tanenbaum

O **Minix** é um sistema operacional Unix-like (semelhante ao UNIX), gratuito e com o código fonte disponível.

1991 - GNU / Linux

Linus Torvalds, criador e principal mantenedor do Kernel Linux.

:) Marcos Monteiro

1993 - Free BSD

O BSD (Berkeley Software Distribution)

:) Marcos Monteiro

PERGUNTAS???

contato@marcosmonteiro.com.br

http://www.marcosmonteiro.com.br

:) Marcos Monteiro

Sistemas Operacionais

Parte 2

Oque é isso mesmo??

:) Marcos Monteiro

Software

Software Básico

Software Aplicativo

Definição

> Sistema Operacional:

principal programa do sistema, que controla todos os recursos do computador (dispositivos físicos e funções de software).

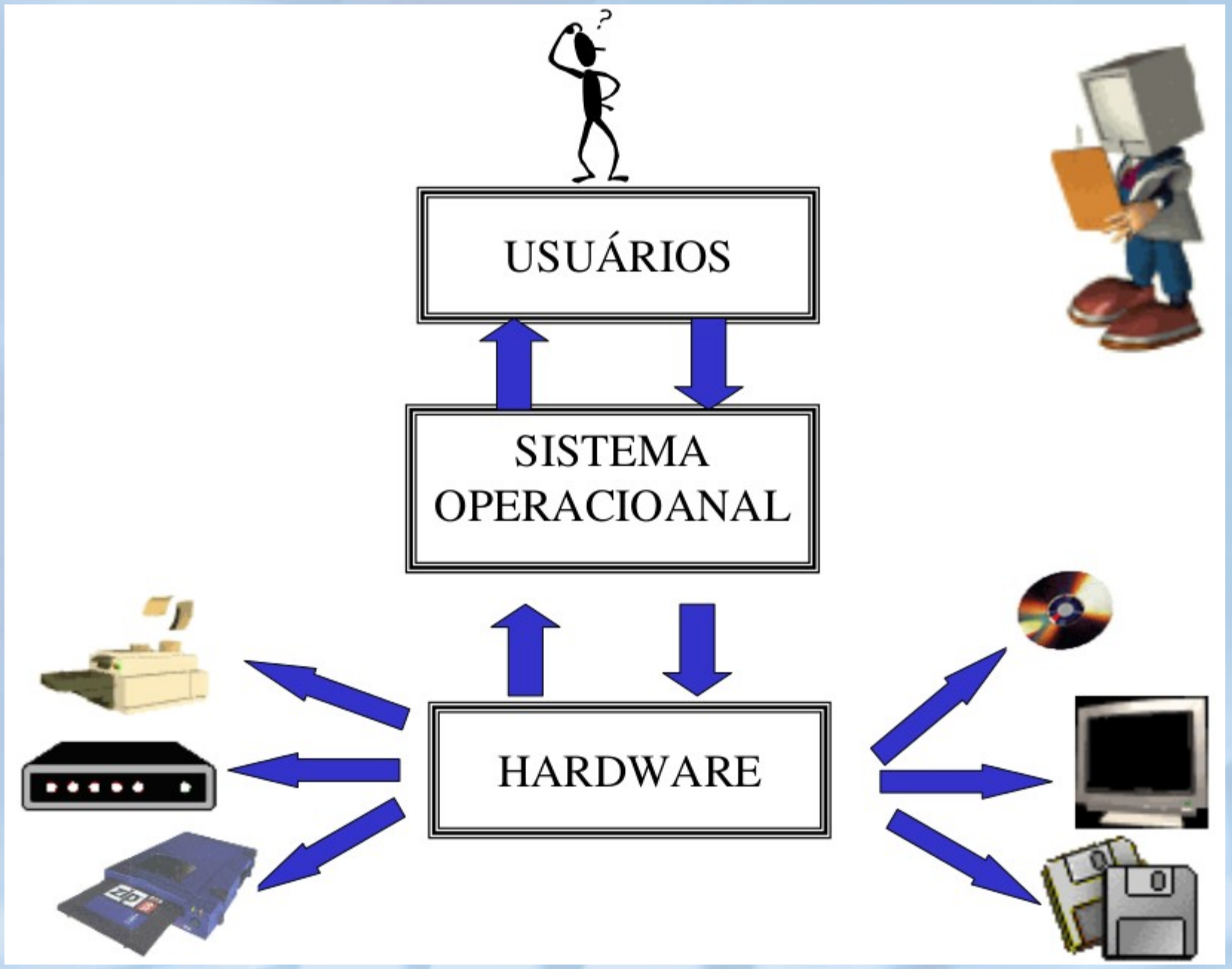
Mais Definição

É um programa de controle do computador. O Sistema Operacional é responsável por alocar recursos de hardware e escalonar tarefas. Ele também deve prover uma interface para o usuário - ele fornece ao usuário uma maneira de acesso aos recursos do computador.

Sobell.

Um Sistema Operacional pode ser definido como um gerenciador dos recursos que compõem o computador (processador, memória, I/O, arquivos, etc). Os problemas centrais que o Sistema Operacional deve resolver são o compar-tilhamento ordenado, a proteção dos recursos a serem usados pelas aplicações do usuário e o interfaceamento entre este e a máquina.”

Stemmer.



:) Marcos Monteiro

Tipos de Sistemas Operacionais

Sistemas Monoprogramáveis / Monotarefa

Sistemas Multiprogramáveis / Multitarefa

Sistemas com Múltiplos Processadores

SISTEMAS MONOPROGRAMÁVEIS / MONOTAREFA

Execução de um único programa (job);

Qualquer outro programa, para ser executado, deveria aguardar o término do programa corrente;

Tipicamente relacionado ao surgimento dos mainframes;

SISTEMAS MULTIPROGRAMÁVEIS / MULTITAREFA

Mais complexos e mais eficientes;

Vários programas dividem os mesmos recursos;

Aumento da produtividade dos seus usuários e a redução de custos;

SISTEMAS COM MÚLTIPLOS PROCESSADORES

Caracterizam por possui duas ou mais UCPs interligadas, trabalhando em conjunto;

– Fortemente Acoplado

• dois ou mais processadores compartilhando uma única memória e controlados por apenas um único SO

– Fracamente Acoplado

• Dois ou mais sistemas de computação interligados, sendo que cada sistema possui o seu próprio SO

Sistemas distribuídos

:) Marcos Monteiro

A computação distribuída, ou sistema distribuído, é uma referência à computação paralela e descentralizada, realizada por dois ou mais computadores conectados através de uma rede, cujo objetivo é concluir uma tarefa em comum.

Definição

- Um sistema distribuído é:
- "coleção de computadores independentes que se apresenta ao usuário como um sistema único e consistente";

Andrew Tanenbaum

- "coleção de computadores autônomos interligados através de uma rede de computadores e equipados com software que permita o compartilhamento dos recursos do sistema: hardware, software e dados"

George Coulouris

Definição

- Assim, a computação distribuída consiste em adicionar o poder computacional de diversos computadores interligados por uma rede de computadores ou mais de um processador trabalhando em conjunto no mesmo computador, para processar colaborativamente determinada tarefa de forma coerente e transparente, ou seja, como se apenas um único e centralizado computador estivesse executando a tarefa. A união desses diversos computadores com o objetivo de compartilhar a execução de tarefas, é conhecida como sistema distribuído.

:) Marcos Monteiro

Modelos de computação distribuída

- **Cliente/Servidor**
  - O *cliente* manda um pedido para o *servidor* e o *servidor* o retorna.
- **Peer-to-peer (P2P)**
  - É uma arquitetura de sistemas distribuídos caracterizada pela descentralização das funções na rede, onde cada nodo realiza tanto funções de servidor quanto de cliente.
- **Objetos Distribuídos**
  - Semelhante ao peer-to-peer, mas com um Middleware "mediador" intermediando o processo de comunicação.

Software

- **Fracamente acoplados**
  - Permitem que máquinas e usuários de um sistema distribuído sejam fundamentalmente independentes e ainda interagir de forma limitada quando isto for necessário, compartilhando discos, impressoras e outros recursos.
- **Fortemente acoplados**
  - provê um nível de integração e compartilhamento de recursos mais intenso e transparente ao usuário caracterizando sistemas operacionais distribuídos.

Cluster

Um cluster, ou aglomerado de computadores, é formado por um conjunto de computadores, que utiliza um tipo especial de sistema operacional classificado como sistema distribuído. Muitas vezes é construído a partir de computadores convencionais (personal computers), os quais são ligados em rede e comunicam-se através do sistema, trabalhando como se fossem uma única máquina de grande porte. Há diversos tipos de cluster. Um tipo famoso é o cluster da classe Beowulf, constituído por diversos nós escravos gerenciados por um só computador.

Tipos de cluster

- Cluster de Alto Desempenho:
  - Também conhecido como cluster de alta performance, ele funciona permitindo que ocorra uma grande carga de processamento com um volume alto de gigaflops em computadores comuns e utilizando sistema operacional gratuito, o que diminui seu custo.

Tipos de cluster

- Cluster de Alta Disponibilidade:
  - São clusters os quais seus sistemas conseguem permanecer ativos por um longo período de tempo e em plena condição de uso. Sendo assim, podemos dizer que eles nunca param seu funcionamento; além disso, conseguem detectar erros se protegendo de possíveis falhas..

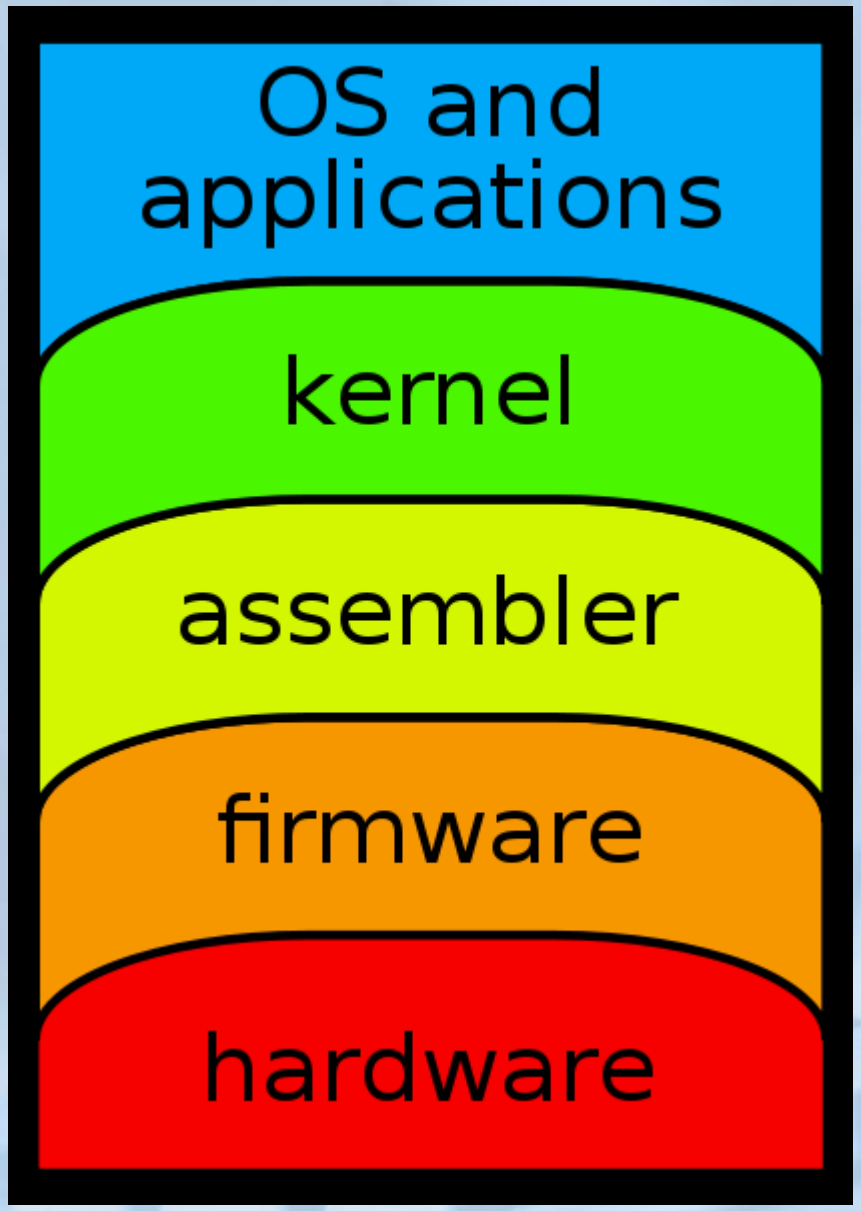
Tipos de cluster

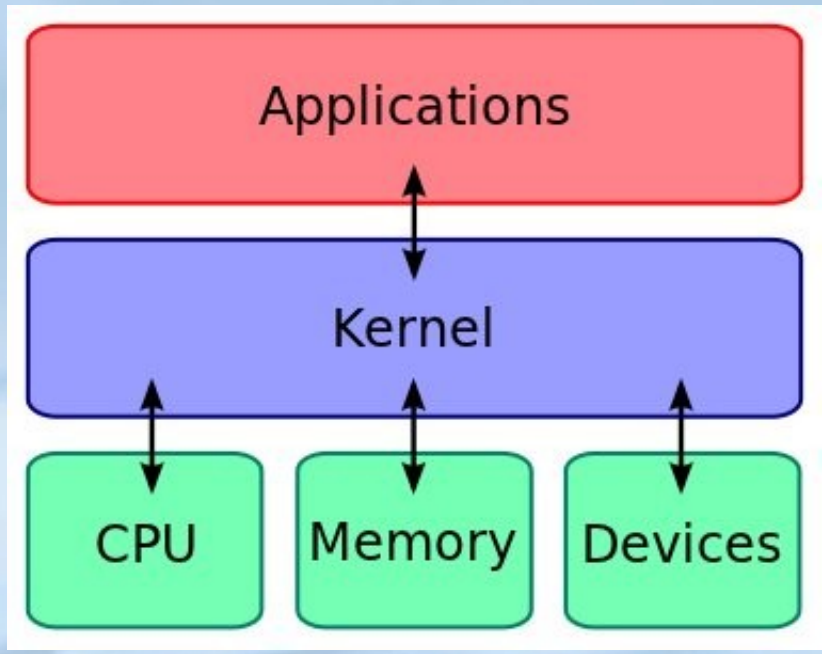
- Cluster para Balanceamento de Carga:
  - Esse tipo de cluster tem como função controlar a distribuição equilibrada do processamento. Requer um monitoramento constante na sua comunicação e em seus mecanismos de redundância, pois se ocorrer alguma falha, haverá uma interrupção no seu funcionamento.

ESTRUTURA DO SO

Formado por um conjunto de rotinas (procedimentos) que oferecem serviços aos usuários do sistema e suas aplicações, bem como a outras rotinas do próprio sistema. Esse conjunto de rotinas é chamado núcleo do sistema ou kernel (Núcleo).

Kernel





:) Marcos Monteiro

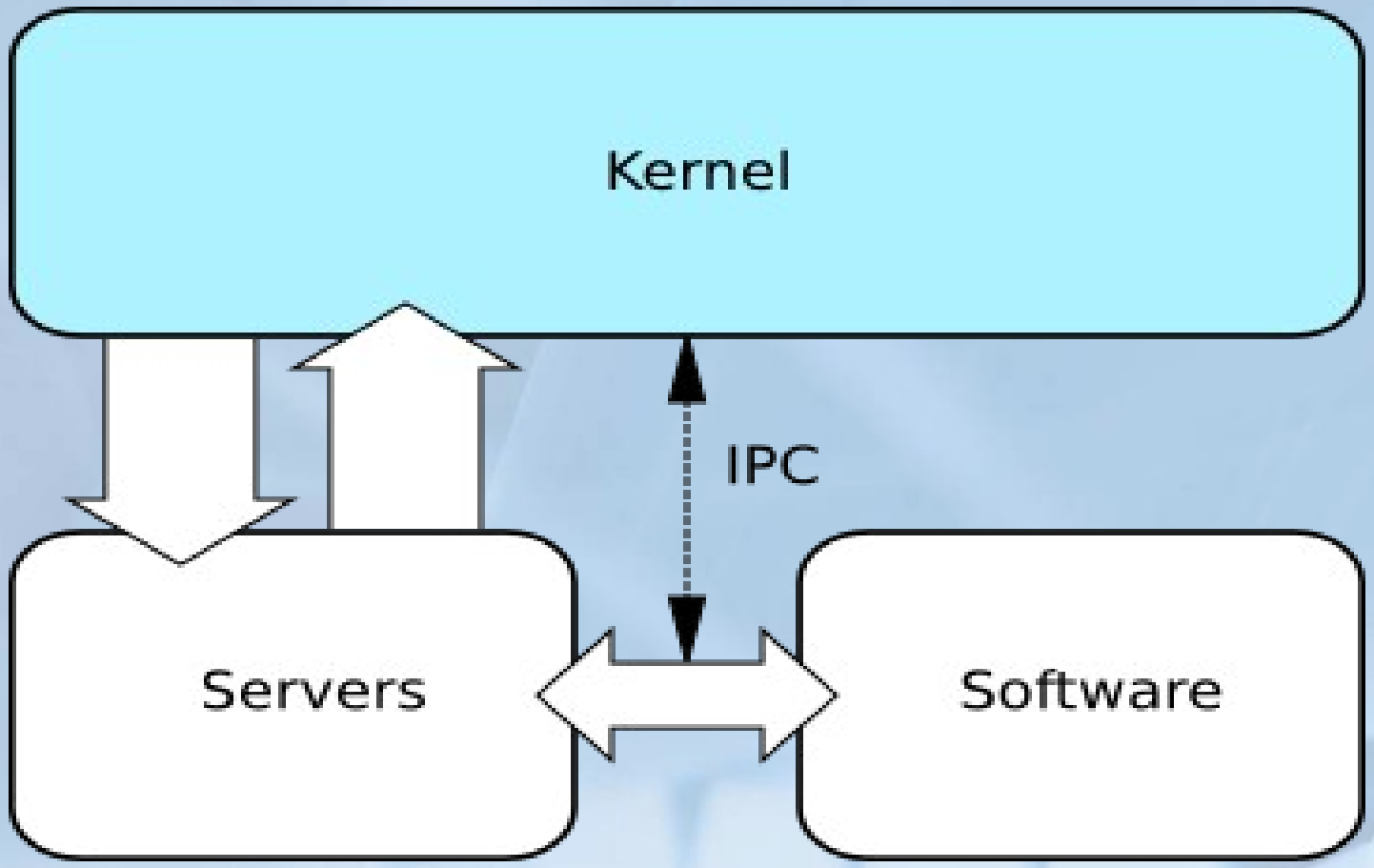
Micro-Kernel ou Kernel Monolítico

Micro-kernel, ou também chamado de microkernel, é uma designação de um Sistema Operacional que possui apenas um núcleo que provê recursos mínimos necessários ao ambiente. Outras funcionalidades são oferecidas através de programas chamados servidores, que se localizam na user-space.

O Micro-Kernel basicamente provê serviços como gerenciamento do espaço de memória, gerência de threads e comunicação entre os processos (IPC – Inter-Process Communication). Serviços como rede, vídeo, são considerados não essenciais, e residem no user-space. A figura a seguir representa a interação entre os softwares que são executados sobre o sistema e sua relação com o kernel.

:) Marcos Monteiro

Micro-Kernel



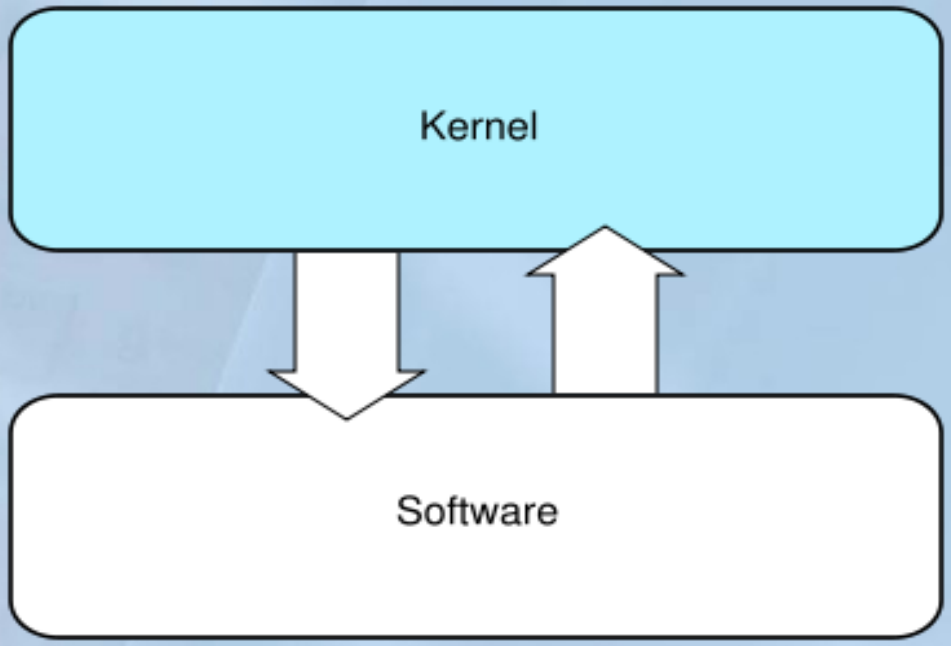
Exemplos de micro-kernel que posso citar são o Hurd e Minix. Utilizando o mesmo conceito, surgiram outras implementações, como os nanokernels e exokernels, sendo que todos utilizam o mesmo princípio minimalista.

:) Marcos Monteiro

Kernel Monolítico

O paradigma do Kernel Monolítico é justamente o oposto do Micro-Kernel. A principal característica do kernel monolítico é permitir que funções como rede, vídeo e acesso a outros periféricos sejam possíveis através do kernel-space. Isso é possível através do uso de módulos. O que significa que um módulo, apesar de não estar no mesmo código do kernel, é executado no espaço de memória do kernel. Sendo assim, apesar de modular, o kernel monolítico continua sendo único e centralizado. Isso pode levar a considerações errôneas sobre o conceito. Segue abaixo uma representação do kernel monolítico.

Kernel Monolítico



Como exemplo desse tipo de arquitetura, posso citar o Linux, BSD e Windows. Em comum com a arquitetura de Micro-Kernels, o kernel define uma interface de alto nível sob o hardware do computador, com um conjunto de primitivas, ou chamadas de sistema para implementação de serviços no sistema operacional.

:) Marcos Monteiro

SYSTEM CALLS

- Mecanismo de proteção ao núcleo do sistema e de acesso aos seus serviços.

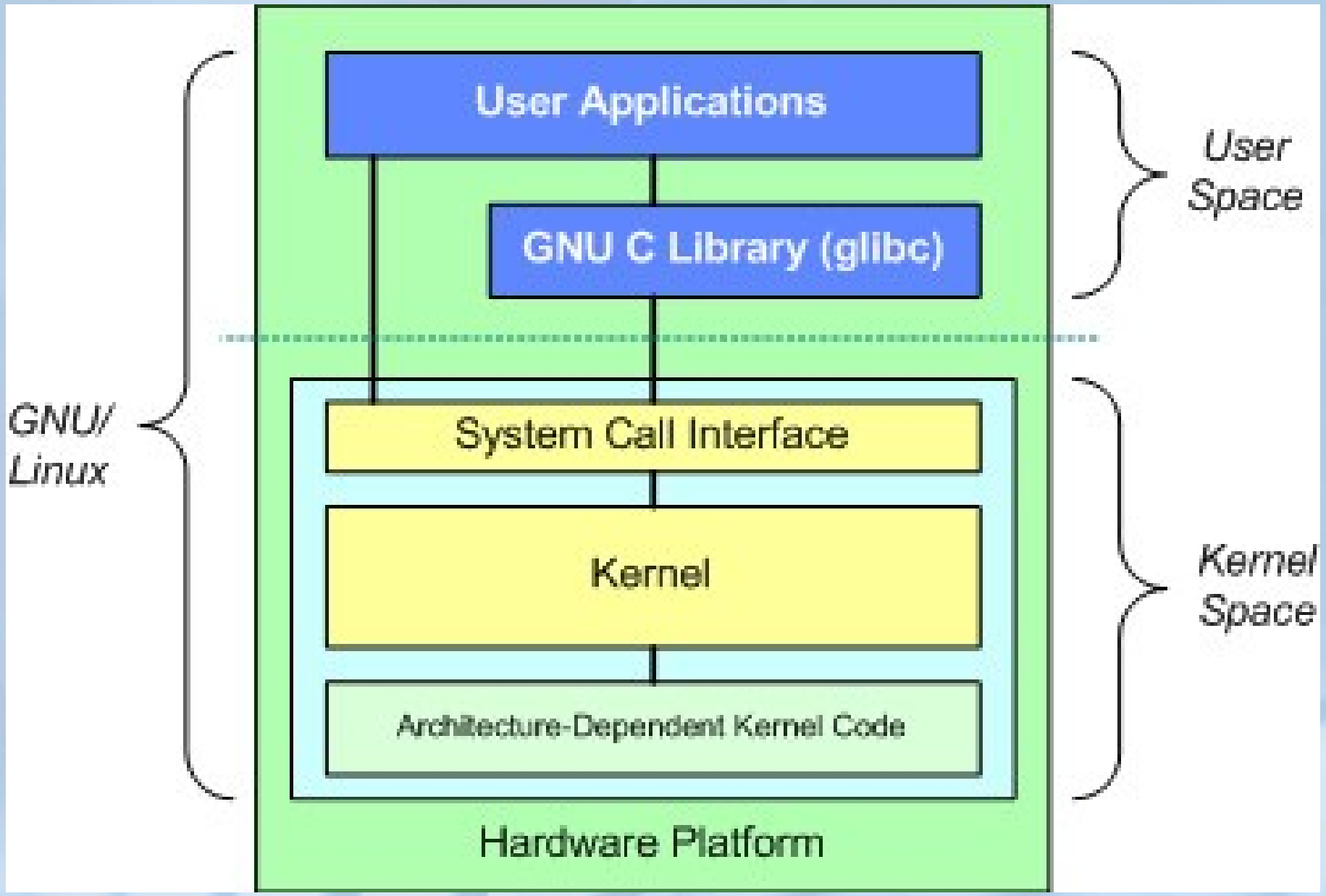
- O usuário (ou aplicação), quando deseja solicitar algum serviço do sistema, realiza uma chamada a uma de suas rotinas (ou serviços) através da system calls (chamadas ao sistema).

MODOS DE ACESSO

• Existem certas instruções que não podem ser colocadas diretamente à disposição das aplicações, pois a sua utilização indevida ocasionaria sérios problemas à integridade do sistema.

• As instruções que têm o poder de comprometer o sistema são conhecidas como instruções privilegiadas (modo kernel), enquanto as instruções não-privilegiadas são as que não oferecem perigo ao sistema.

• Registrador da UCP, que indica o modo de acesso corrente.

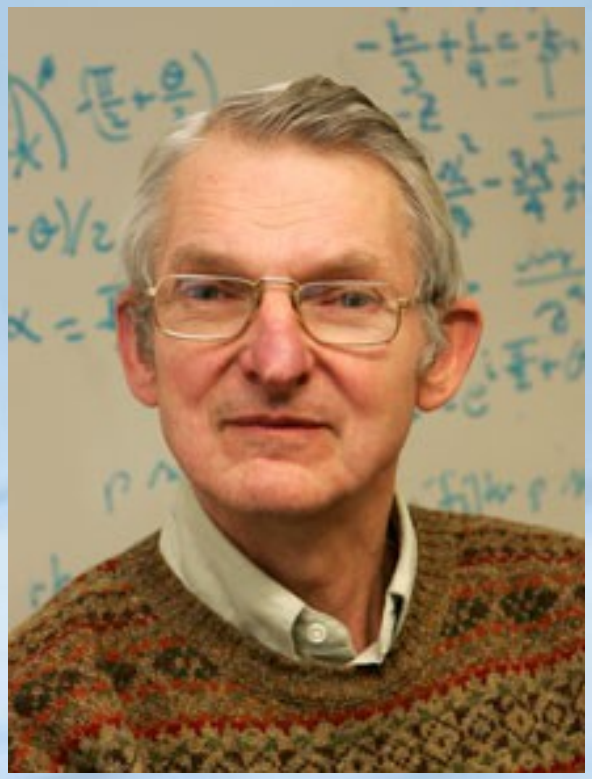


:) Marcos Monteiro

Arquitetura de um Unix-like

Pipes and Filters

(Dutos e filtros)



Douglas Mcilroy

:) Marcos Monteiro

Dutos e filtros

- Esse padrão oferece uma estrutura para sistemas que processam fluxos de dados
  - Cada passo de processamento é encapsulado em um filtro
  - O dado é passado pelos dutos entre filtros adjacentes
  - Recombinação de filtros permite a construção de famílias de sistemas relacionados

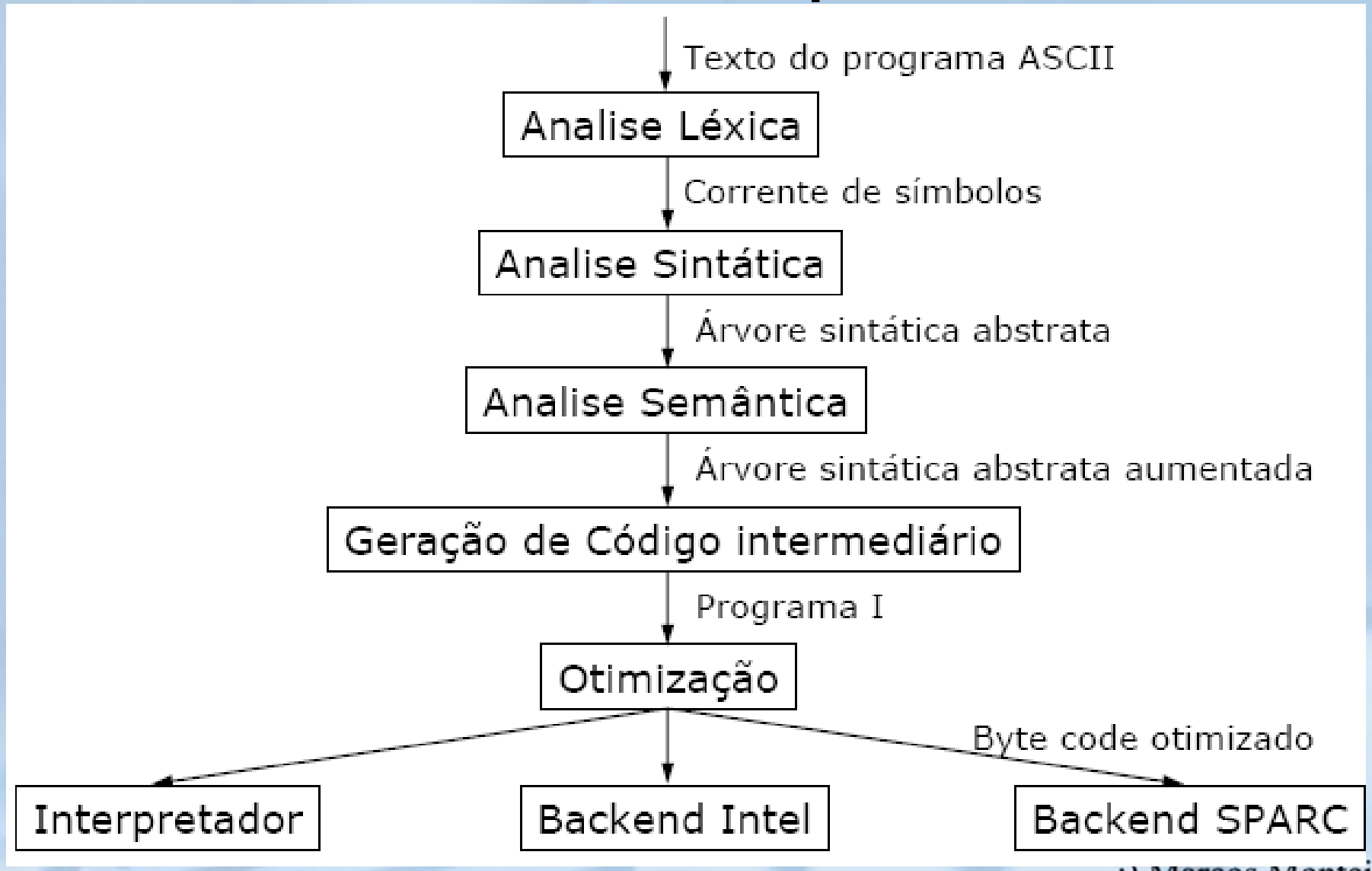
Componentes

- Filtro
- Data source
- Data sink
- Duto

Um exemplo

- Construção de um compilador portável para uma linguagem L
  - L é compilada para uma linguagem intermediária I que roda em uma máquina virtual VM
  - VM será implementada por um interpretador ou backends específicos para diferentes plataformas
  - Um backend irá traduzir o código para instruções de máquina de um processador específico para melhor desempenho

Um exemplo


Texto do programa ASCII
Analise Léxica
Corrente de símbolos
Analise Sintática
Árvore sintática abstrata
Analise Semântica
Árvore sintática abstrata aumentada
Geração de Código intermediário
Programa I
Otimização
Byte code otimizado
Interpretador
Backend Intel
Backend SPARC

Problema

- Imagine que você está construindo um sistema que deve processar ou transformar um stream de dados de entrada
  - A implementação do sistema como componente único não é possível por diversas razões
  - O sistema deve ser implementado por várias pessoas,
  - A tarefa pode ser decomposta naturalmente em diversos passos de processamento, e os passos de processamento podem ser modificados, reordenados ou reutilizados em outros contextos

Solução

- A arquitetura de Dutos e Filtros divide as tarefas do sistema em diversos passos de processamento seqüencial

- Os passos estão conectados pelo fluxo de dados através do sistema: a saída de um passo é a entrada do passo seguinte

- Cada passo é implementado por um componente chamada filtro

- Um filtro consome e entrega dados de forma incremental

Solução

- A entrada para o sistema é feita através de uma fonte de dados (data source), por exemplo, um arquivo texto
- A saída flui para um sorvedouro de dados (data sink), por exemplo, um arquivo, programa, etc.
- A fonte de dados, os filtros e a saída de dados estão conectados seqüencialmente por dutos (dutos)
- Cada duto implementa o fluxo de dados entre passos de processamento adjacentes.
- A seqüência de filtros combinados através de dutos é chamada de *processing pipeline*

Estrutura

Classe	**Colaboradores**
Filter	• Pipe
Responsabilidades:	
• Pega o dado de entrada • Processa uma função sobre o dado de entrada • Fornece o dado de saída	

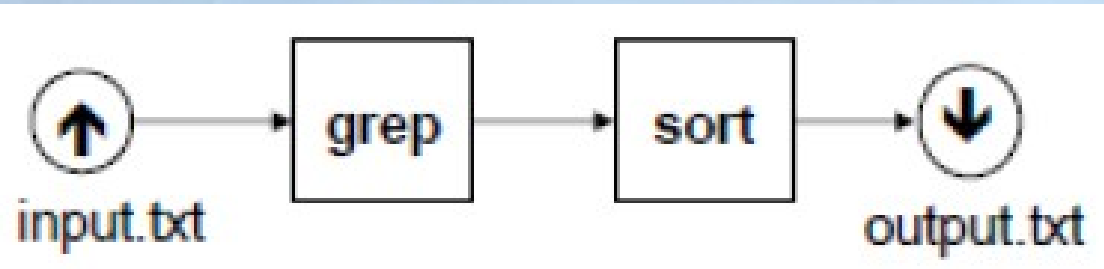
Classe	**Colaboradores**
Pipe	• Data Source • Data Sink • Filter
Responsabilidades:	
• Transferência de dados • Armazenamento temporário de Dados • Sincronização de atividades vizinhas	

Classe	**Colaboradores**
Data Source	• Pipe
Responsabilidades:	
• Fornece entrada para a linha de processamento	

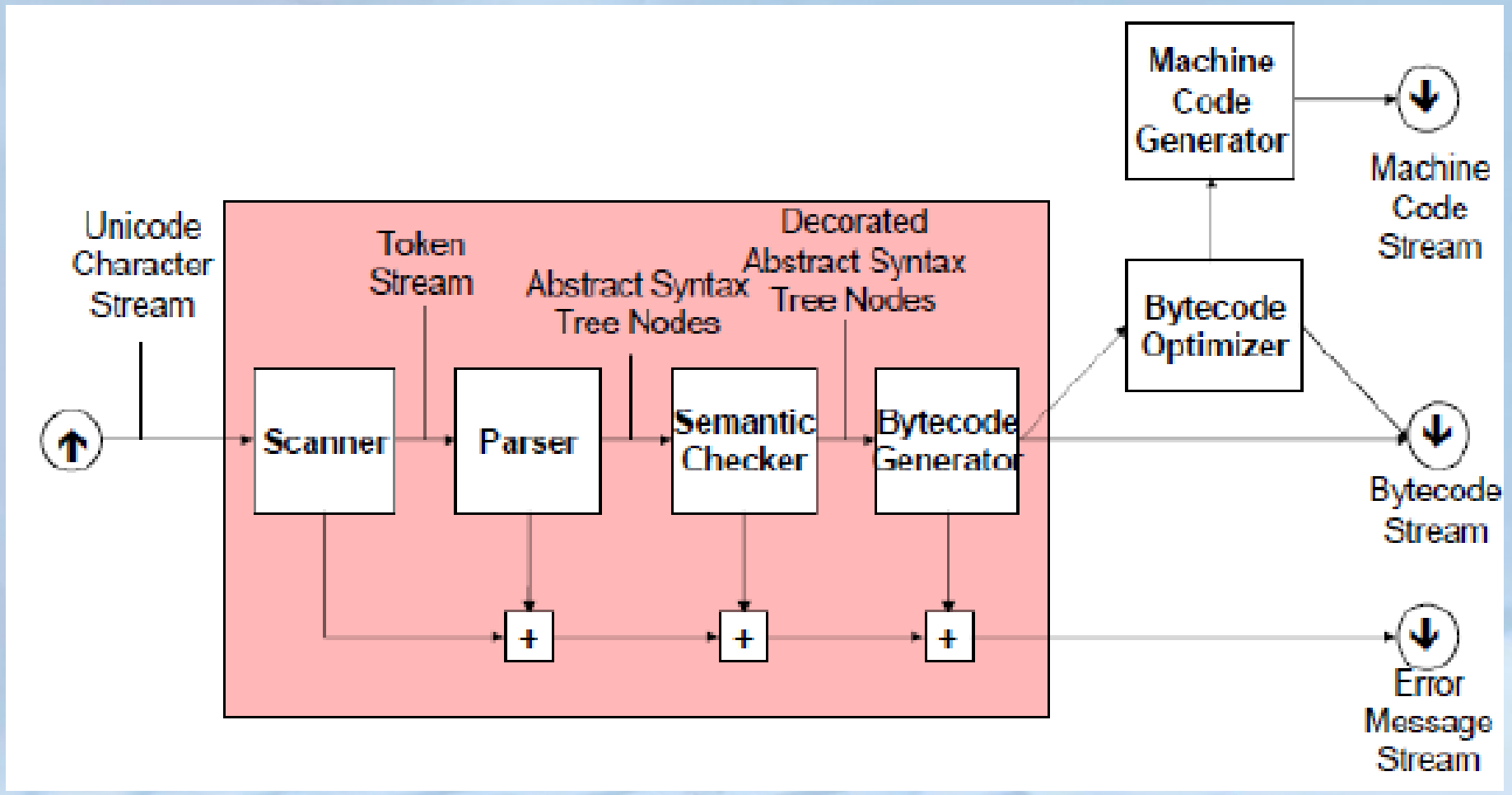
Classe	**Colaboradores**
Data Sink	• Pipe
Responsabilidades:	
• Consume a saída.	

Dutos e filtros

- Sistemas operacionais e compliladores são os melhores exemplos de arquitetura dutos e filtros
- Exemplo SO
  - Unix shell:
    - cat input.txt | grep “text” | sort > output.txt



Exemplo compilador



Dutos e filtros – implementação

- Divida a tarefa do sistema em uma seqüência de passos de processamento
  - Cada estágio deve depender apenas da saída do seu predecessor direto
  - Todos os estágios são conectados pelo fluxo de dados
- Defina o formato dos dados que serão passados por cada duto
- Decida como implementar cada conexão de duto
- Projete implemente os filtros
- Projete o tratamento de erros