

INSTITUTO FEDERAL
Sul-rio-grandense

Câmpus
Santana do Livramento

Sistemas Operacionais

Aula 06

Prof. Victor Machado Alves

victor.alves@ifsul.edu.br

Introdução

- Aplicações precisam
 - Armazenar e recuperar informações
- Enquanto o processo executa, onde ficam informações?
 - Informações ficam dentro do seu espaço de endereçamento
 - Para algumas aplicações isto é suficiente
 - Para outras não...
- Quando a informação fica apenas no espaço de endereçamento do processo, o que acontece quando termina?
 - Após sua finalização, informações são perdidas
 - Muitas aplicações necessitam que dados sejam mantidos

Introdução

- Outra questão:
 - Necessidade de múltiplos processos tenham acesso à informação ao mesmo tempo
 - Se esta informação estiver dentro do espaço de endereçamento do processo A
 - Processos B, C e D não conseguirão ter acesso
- Solução:
 - Tornar a informação independente de qualquer processo

Introdução

- Requisitos para armazenamento da informação por longo prazo:
 - Possibilidade de armazenar uma quantidade grande de informação
 - Informação deve existir após o término do processo
 - Múltiplos processos devem ser capazes de acessar a informação de maneira concorrente

Introdução

- Especialmente quando os sistemas tornam-se grandes e abrigam uma quantidade igualmente expressiva de usuários, algumas questões devem ser respondidas:
 - Como encontrar uma informação que foi armazenada?
 - Como impedir que um usuário tenha acesso aos dados de outro usuário?
 - Como saber quais blocos estão livres para uso?
- Sistema Operacional
 - Resolve a situação através da criação de uma abstração (assim como o fez em relação ao processador e memória)
 - Abstração de um arquivo

Introdução

- Arquivos
 - Unidades lógicas de informação
 - Unidade de disco pode conter milhares de arquivos
 - Arquivos são como o espaço de endereçamento
 - Mas usados para modelar o disco e não a memória
 - Persistência
 - Término de processos não afeta informação guardada

Arquivos

- É um mecanismo de abstração
- Meio de armazenar e recuperar informação no disco
- Isolar do usuário os detalhes de como e onde foi realmente armazenado

Nomeação

- Quando um processo cria um arquivo o mesmo precisa ser nomeado
- Outros processos podem acessar este arquivo através do nome
- Muitos SOs permitem nomes de arquivos com duas partes separadas por um ponto
 - Nome do arquivo
 - Extensão
 - Indica algo sobre o arquivo

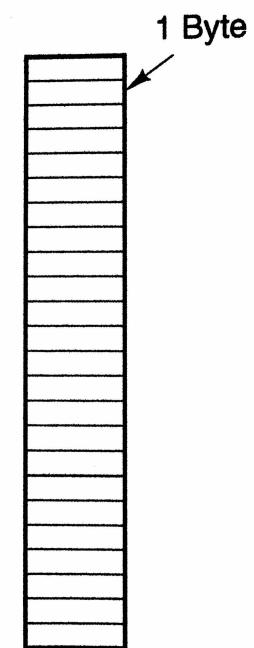
Nomeação

-

Extensão	Significado
.bak	Cópia de segurança
.c	Código-fonte de programa em C
.gif	Imagem no formato <i>Graphical Interchange Format</i>
.hlp	Arquivo de ajuda
.html	Documento em HTML
.jpg	Imagen codificada segundo padrões JPEG
.mp3	Música codificada no formato MPEG (camada 3)
.mpg	Filme codificado no padrão MPEG
.o	Arquivo objeto (gerado por compilador, ainda não ligado)
.pdf	Arquivo no formato PDF (<i>Portable Document File</i>)
.ps	Arquivo PostScript
.tex	Entrada para o programa de formatação TEX
.txt	Arquivo de texto
.zip	Arquivo compactado

Estrutura de Arquivos

- Sistemas operacionais como versões do UNIX e Windows enxergam os arquivos como sequência de bytes
 - Não importa o que realmente o arquivo contém
 - São vistos como sequência de bytes



Tipos de Arquivos

- UNIX e Windows
 - Arquivos regulares e diretórios
 - Arquivos regulares
 - Possuem informação do usuário
 - Diretórios
 - Arquivos do sistema que definem a estrutura do sistema de arquivos
- UNIX
 - Arquivos especiais de caracteres
 - Relacionados a dispositivos de entrada e saída
 - Arquivos especiais de bloco
 - Usados para modelar discos

Tipos de Arquivos

- Arquivos regulares
 - Em geral arquivos ASCII ou binários
 - ASCII
 - Apresentam a vantagem de poderem ser editados, mostrados e impressos
 - Programas podem ser facilmente conectados se usarem arquivos ASCII
 - Lembram-se do uso de pipes?
 - Binário
 - Possuem uma estrutura interna, conhecida pelos programas que os utilizam
 - Imprimir este tipo de arquivo resultaria em um resultado não comprehensível

Atributos de Arquivos

- Arquivo possui
 - Nome e seus dados
 - Outras informações:
 - Chamado de atributos ou metadados
 - Exemplo:
 - Data e hora de modificação
 - Tamanho do arquivo

Atributos de Arquivos

Atributo	Significado
Proteção	Quem tem acesso ao arquivo e de que modo
Senha	Necessidade de senha para acesso ao arquivo
Criador	ID do criador do arquivo
Proprietário	Proprietário atual
Flag de somente leitura	0 para leitura/escrita; 1 para somente leitura
Flag de oculto	0 para normal; 1 para não exibir o arquivo
Flag de sistema	0 para arquivos normais; 1 para arquivos do sistema
Flag de arquivamento	0 para arquivos com backup; 1 para arquivos sem backup
Flag de ASCII/binário	0 para arquivos ASCII; 1 para arquivos binários
Flag de acesso aleatório	0 para acesso somente sequencial; 1 para acesso aleatório
Flag de temporário	0 para normal; 1 para apagar o arquivo ao sair do processo
Flag de travamento	0 para destravados; diferente de 0 para travados
Tamanho do registro	Número de bytes em um registro
Posição da chave	Posição da chave em cada registro
Tamanho do campo-chave	Número de bytes no campo-chave
Momento de criação	Data e hora de criação do arquivo
Momento do último acesso	Data e hora do último acesso do arquivo
Momento da última alteração	Data e hora da última modificação do arquivo
Tamanho atual	Número de bytes no arquivo
Tamanho máximo	Número máximo de bytes no arquivo

Diretórios

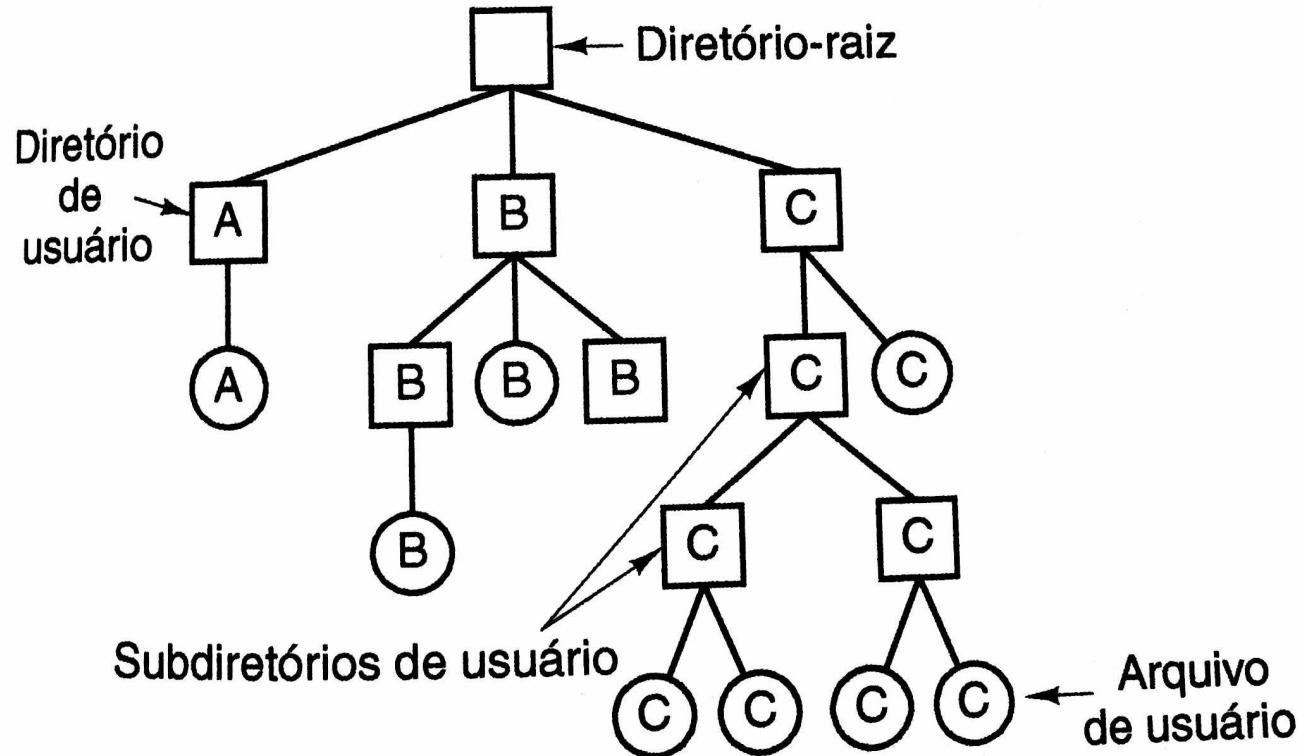
- Para controlar o sistema de arquivos
 - Uso de diretórios ou pastas

Sistemas de Diretórios Hierárquicos

- Necessidade de agrupar os arquivos relacionados em um mesmo local
- Cada usuário pode ter quantos diretórios precisar
- Hierarquia alcançada através de uma árvore de diretórios

Sistemas de Diretórios Hierárquicos

-



Nomes de Caminhos

- Quando os arquivos são organizados através de árvores
 - Necessário uma forma para especificar o nome dos arquivos
 - Duas formas
 - Nome de Caminho Absoluto
 - Caminho entre o diretório raiz e o arquivo
 - Nome de Caminho Relativo
 - Diretório de trabalho/Diretório atual
 - Caminhos que não começem pelo diretório raiz
 - Considera-se que começem pelo caminho relativo

Nomes de Caminhos

- Exemplos:

Se o caminho atual (pasta atual) for /usr/ast/

Ambas soluções possuem o mesmo resultado:

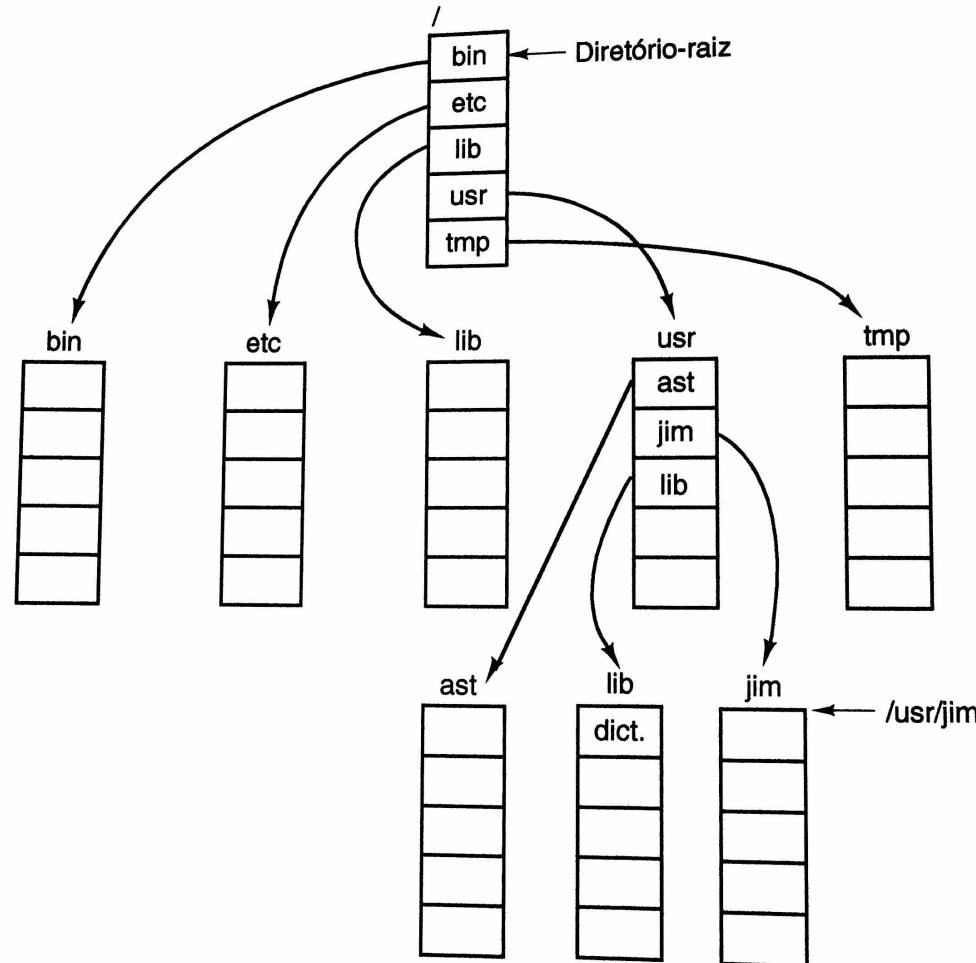
```
cp /usr/ast/caixapostal /usr/ast/caixapostal.bak
```

```
cp caixapostal caixapostal.bak
```

Nomes de Caminhos

- Duas entradas especiais em cada diretório
 - ..
 - Ponto
 - ...
 - Pontoponto
- Exemplo:
 - Processo tem como diretório /usr/ast
 - .. (pontoponto) Usado para subir um nível na árvore
 - Copiar arquivo /usr/lib/dicionario para o próprio diretório usando o comando:
 - cp ../lib/dicionario .

Nomes de Caminhos



Implementação de Arquivos

- Importante questão no armazenamento de arquivos
 - Manutenção e controle de quais blocos de discos estão relacionados a quais arquivos
 - Diferentes métodos:
 - Alocação contígua
 - Alocação por lista encadeada
 - Alocação por lista encadeada usando uma tabela na memória
 - i-nodes

Alocação contígua

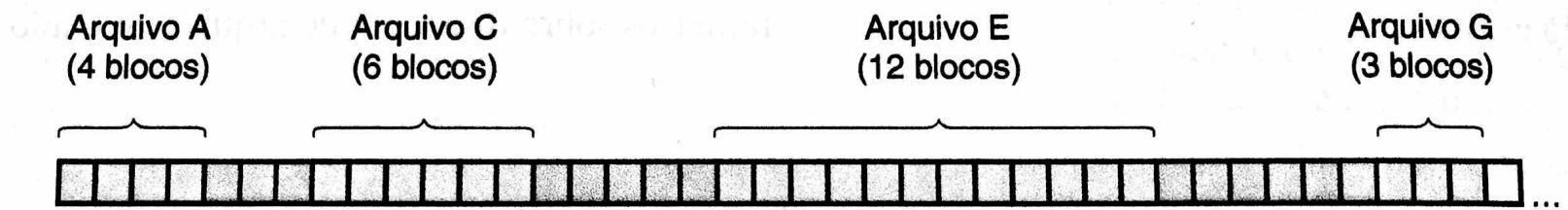
- Mais simples
 - Disco com blocos de 1KB
 - Arquivo com 50 KB
 - Precisaria de... 50 blocos
 - Com blocos de 2 KB
 - O mesmo arquivo de 50 KB
 - Precisaria de... 25 blocos

Alocação contígua

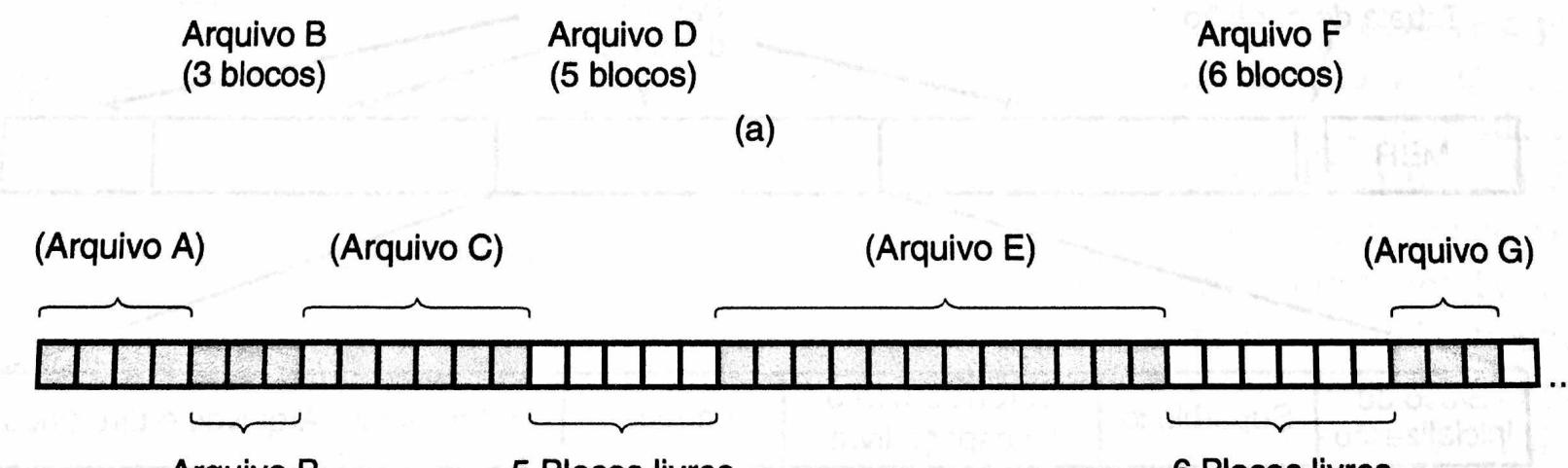
- Vantagens:
 - Simplicidade de implementação
 - Para encontrar um arquivo, necessário saber endereço do primeiro bloco e a quantidade de blocos
 - Leitura de um arquivo tem desempenho satisfatório
 - Blocos estão juntos
- Desvantagem
 - Com o tempo o disco sofre com a fragmentação

Alocação contígua

-



(a)



(b)

Alocação contígua

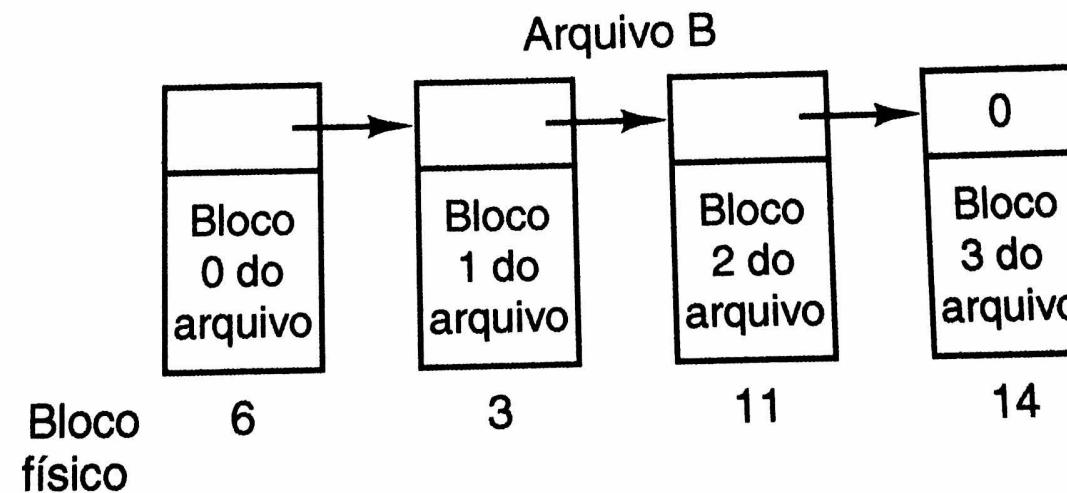
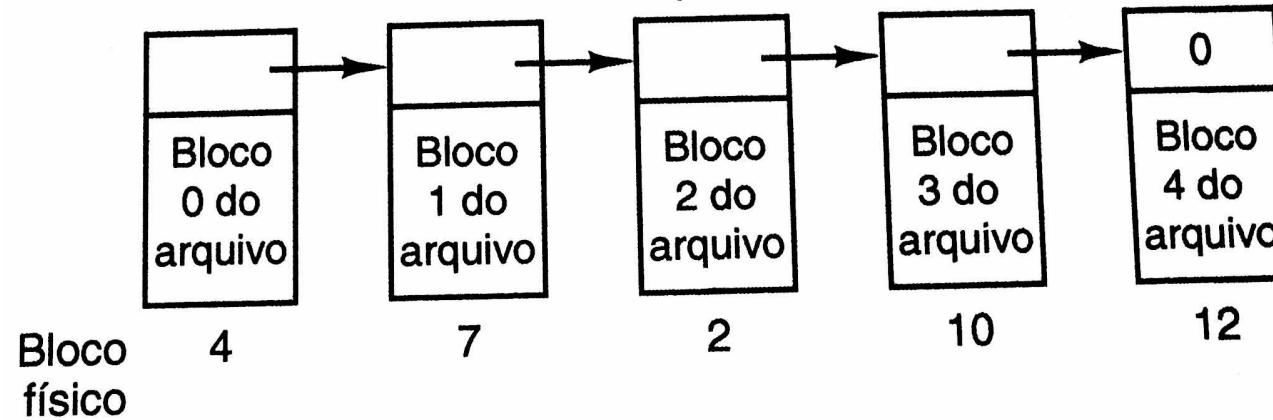
- Exemplo de uso:
 - CD-ROMs

Alocação por lista encadeada

- Cada arquivo
 - Lista encadeada de blocos de disco
 - Primeira palavra de cada bloco usada como ponteiro para o próximo
 - Sem o problema da fragmentação
 - Suficiente armazenar o endereço do primeiro bloco do arquivo
 - Os demais serão encontrados através de ponteiros
- Desvantagem
 - A leitura não terá o mesmo desempenho que a alocação contígua
 - Bloco para dados é reduzido, pois o ponteiro necessita de alguns bytes

Alocação por lista encadeada

-

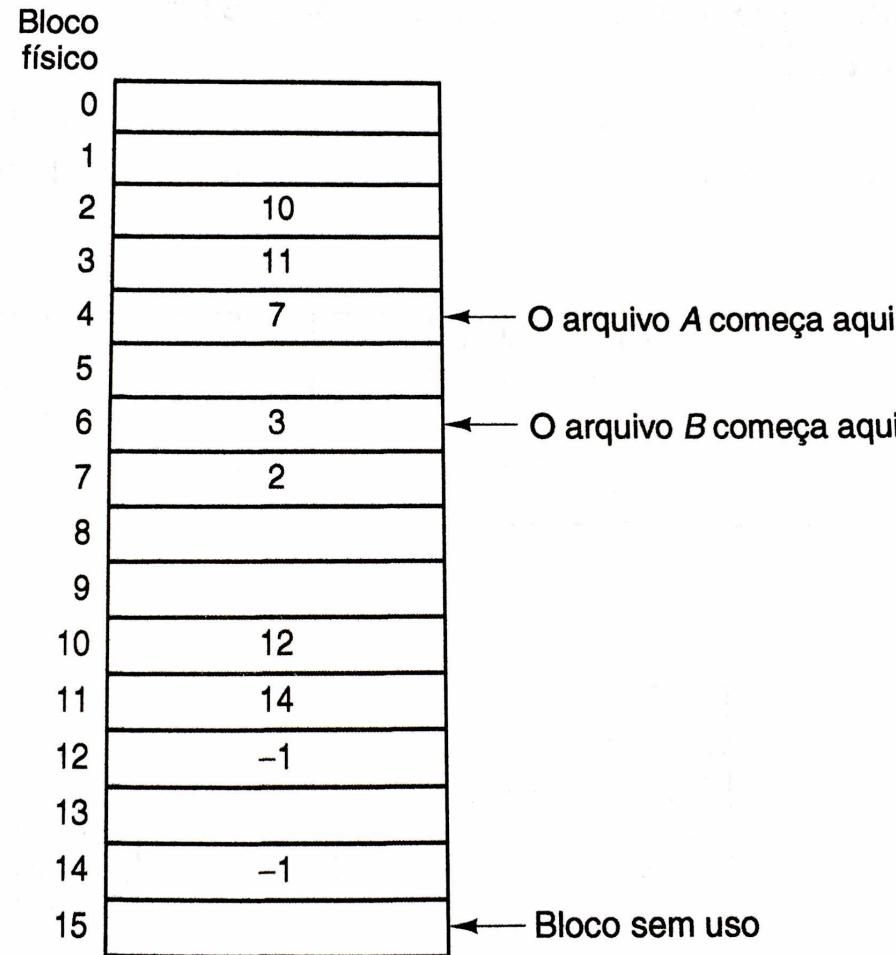


Alocação por lista encadeada usando uma tabela na memória

- Uso de uma tabela na memória principal
- Blocos passam a ser utilizados totalmente para dados
- File Allocation Table (FAT)
 - Tabela de Alocação de Arquivos
- Acesso aleatório fica mais fácil
 - Mapeamento está diretamente na memória
 - Sem precisar referenciar do disco para encontrar a próxima posição
- Principal desvantagem
 - Necessidade da tabela estar, totalmente, na memória

Alocação por lista encadeada usando uma tabela na memória

-

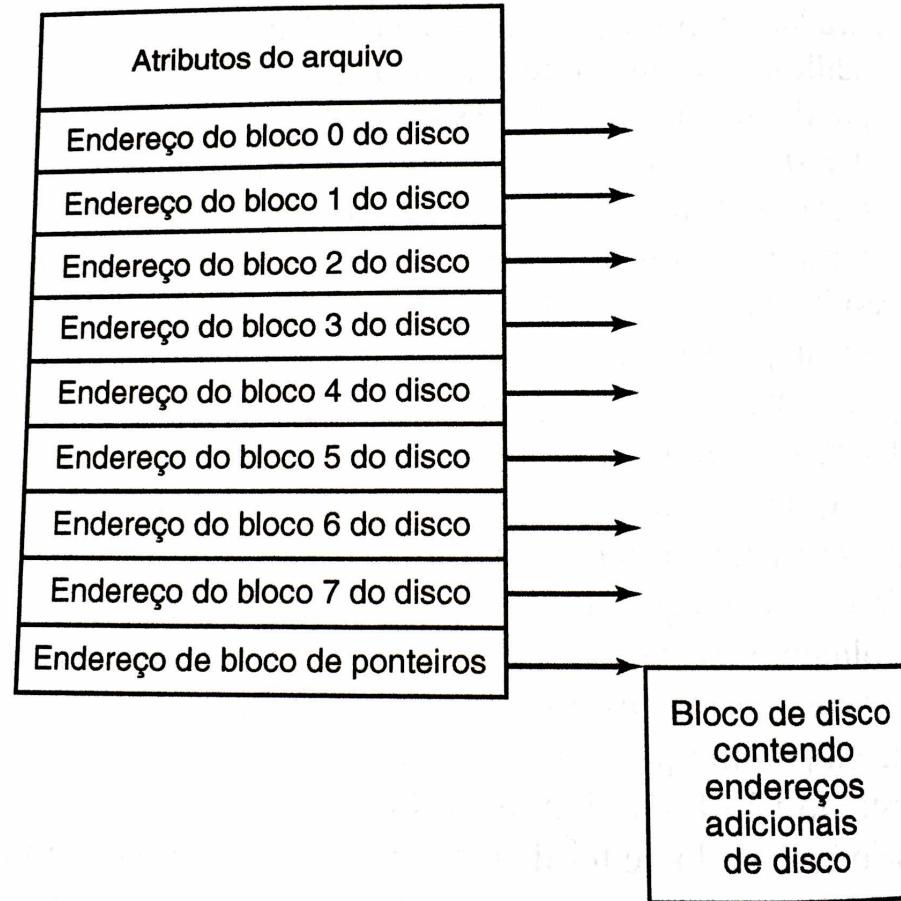


i-nodes

- Associar a cada arquivo uma estrutura de dados chamada i-node (index-node)
 - Relaciona os atributos e
 - Endereços em disco dos blocos de arquivo
- Vantagem sobre o métodos anterior
 - i-node só precisa estar na memória quando o arquivo correspondente estiver aberto
- E se o arquivo aumentar além do limite esperado?
 - Último endereço pode ser usado como ponteiro para mais endereços de blocos que o arquivo possa utilizar

i-nodes

-



Sistema de Arquivos journaling

- Manter registro sobre o que o sistema de arquivos irá fazer antes que efetivamente faça
 - Se falhar, antes de executar a tarefa
 - Possibilidade de Retomar o trabalho
 - Exemplos:
 - NTFS, ext3, ReiserFS

Sistema de Arquivos journaling

- Exemplo:
 - Remoção de um arquivo
 - Remove o arquivo do diretório
 - Libera o i-node para o conjunto de i-nodes livres
 - Coloca todos os blocos do disco para o conjunto de blocos livres do disco
 - Sistema escreve em um arquivo a descrição das 3 operações
 - Após operação realmente acontecer
 - Arquivo é excluído

-

Diretório-raiz

1	.
1	..
4	bin
7	dev
14	lib
9	etc
6	usr
8	tmp

Procurar usr
resulta no
i-node 6

I-node 6
é para /usr

Mode	.
size	.
times	.

132

Bloco 132
é o diretório /usr

6	.
1	..
19	dick
30	erik
51	jim
26	ast
45	bal

I-node 26
é para
/usr/ast

Mode	.
size	.
times	.

406

Bloco 406
é o diretório /usr/ast

26	.
6	..
64	grants
92	books
60	mbox
81	minix
17	src

I-node 26
diz que
/usr/ast
está no i-node
26

I-node 26
diz que
/usr/ast/mbox
está no i-node
60

```
● Error while reading over extent tree in inode 16472: Corrupt extent header
Clear inode? yes

Inode 16472, i_blocks is 8288, should be 0. Fix? yes

Inode 31105 was part of the orphaned inode list. FIXED.
Inode 31105 has a bad extended attribute block 524288. Clear? yes

Inode 31106, i_size is 410, should be 36864. Fix? yes

Inode 31106, i_blocks is 136, should be 72. Fix? yes

Inode 31107 has a extra size (16412) which is invalid
Fix? yes

Deleted inode 31108 has zero dtime. Fix? yes

Running additional passes to resolve blocks claimed by more than one inode...
Pass 1B: Rescanning for multiply-claimed blocks
Multiply-claimed block(s) in inode 2316: 54534
Illegal block number passed to ext2fs_test_block_bitmap #2147483648 for multiply claimed b
lock map
Multiply-claimed block(s) in inode 2324: 54492 54493 54494 54495 54496 54497 54498 54499 5
```

Ubuntu 15.10 sk-desktop tty1

```
sk-desktop login: [  31.620380] EXT4-fs error (device sda1): ext4_lookup:1584:  
inode #1057289: comm rm: deleted inode referenced: 51  
[  31.627403] EXT4-fs error (device sda1): ext4_lookup:1584: inode #1057289: co  
mm rm: deleted inode referenced: 51  
[  45.879111] EXT4-fs error (device sda1): ext4_lookup:1584: inode #162329: com  
m lightdm: deleted inode referenced: 131233  
[  45.920311] EXT4-fs error (device sda1): ext4_lookup:1584: inode #1057289: co  
mm rm: deleted inode referenced: 51  
[  45.921552] EXT4-fs error (device sda1): ext4_lookup:1584: inode #1057289: co  
mm rm: deleted inode referenced: 51  
[  58.591387] EXT4-fs error (device sda1): ext4_lookup:1584: inode #162329: com  
m lightdm: deleted inode referenced: 131233  
[  58.629158] EXT4-fs error (device sda1): ext4_lookup:1584: inode #1057289: co  
mm rm: deleted inode referenced: 51  
[  58.630111] EXT4-fs error (device sda1): ext4_lookup:1584: inode #1057289: co  
mm rm: deleted inode referenced: 51  
[  71.153894] EXT4-fs error (device sda1): ext4_lookup:1584: inode #162329: com  
m lightdm: deleted inode referenced: 131233  
[  71.188496] EXT4-fs error (device sda1): ext4_lookup:1584: inode #1057289: co  
mm rm: deleted inode referenced: 51  
[  71.189489] EXT4-fs error (device sda1): ext4_lookup:1584: inode #1057289: co  
mm rm: deleted inode referenced: 51
```

```
●
Inode 263986 was part of the orphaned inode list.  FIXED.
Inode 263987 was part of the orphaned inode list.  FIXED.
Inode 264058 was part of the orphaned inode list.  FIXED.
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
Block bitmap differences: -2386430 -3297505
Fix<y>? yes
Free blocks count wrong for group #72 (6801, counted=6802).
Fix<y>? yes
Free blocks count wrong for group #100 (25045, counted=25046).
Fix<y>? yes
Free blocks count wrong (1644944, counted=1566127).
Fix<y>? yes
Inode bitmap differences: -(263984--263987) -264058
Fix<y>? yes
Free inodes count wrong for group #32 (83, counted=88).
Fix<y>? yes
Free inodes count wrong (753030, counted=751119).
Fix<y>? yes

/dev/sda1: ***** FILE SYSTEM WAS MODIFIED *****
/dev/sda1: 362993/1114112 files (0.7% non-contiguous), 2890065/4456192 blocks
(initramfs) _
```

Entrada/Saída

- SO deve controlar todos os dispositivos de E/S
 - E/S = entrada/saída
 - Tarefas como:
 - Enviar comandos para dispositivos
 - Interceptar interrupções
 - Tratar erros
 - Oferecer interface entre os dispositivos e o restante do sistema
 - Simples e fácil de usar

Hardware de E/S

- Aqui, o foco está em como o hardware é programado
 - Como funciona
 - Qual interface para interação

Hardware de E/S

- Dispositivos de E/S
 - Divididos em 2 categorias
 - Dispositivos de blocos
 - Dispositivos de caracteres

Hardware de E/S

- Dispositivos de E/S
 - Divididos em 2 categorias
 - Dispositivos de blocos
 - Armazena informação em blocos de tamanho fixo
 - Cada um com endereçamento próprio
 - Blocos variam de 512 bytes a 32.768 bytes
 - Transferências estão em unidades de um ou mais blocos inteiros
 - Cada bloco pode ser lido ou escrito independente dos outros
 - Exemplos?
 - Discos rígidos, CD-ROMS, pendrives

Hardware de E/S

- Dispositivos de E/S
 - Divididos em 2 categorias
 - Dispositivos de caracteres
 - Envia/recebe fluxo de caracteres
 - Não considera estrutura de blocos
 - Não é endereçável
 - Exemplos?
 - Impressora, interface de rede, mouse

Hardware de E/S

- Dispositivos de E/S
 - Podem apresentar variação de velocidade
 - Impacto no software
 - Deve funcionar com taxas de transferência de dados de diferentes ordens

Dispositivo	Taxa de dados
Teclado	10 bytes/s
Mouse	100 bytes/s
Modem 56 K	7 KB/s
Scanner	400 KB/s
Filmadora camcorder digital	3,5 MB/s
Rede sem fio 802,11g	6,75 MB/s
CD-ROM 52x	7,8 MB/s
Fast Ethernet	12,5 MB/s
Cartão flash compacto	40 MB/s
FireWire (IEEE 1394)	50 MB/s
USB 2.0	60 MB/s
Padrão SONET OC-12	78 MB/s
Disco SCSI Ultra 2	80 MB/s
Gigabit Ethernet	125 MB/s
Drive de disco SATA	300 MB/s
Fita Ultrium	320 MB/s
Barramento PCI	528 MB/s

Hardware de E/S

- Controladores de dispositivos
 - Unidades de E/S
 - Geralmente compostos por um componente **mecânico** e um componente **eletrônico**
 - Componente eletrônico
 - Chamado de Controlador do dispositivo (adaptador)
 - Exemplo:
 - Placa controladora
 - Componente mecânico
 - Dispositivo

Hardware de E/S

- Controladores de dispositivos
 - Placa controladora
 - Geralmente possui interface que permite a interligação com o dispositivo em questão
 - São capazes de controlar vários dispositivos iguais
 - Se a interface entre controlador e dispositivo for padrão (oficial)
 - Empresas podem desenvolver controladores e/ou dispositivos que utilizem tal interface
 - Exemplo:
 - Empresas desenvolvem controladores de disco compatíveis com interfaces IDE, SATA, USB

Hardware de E/S

- Controladores de dispositivos
 - Interface entre controlador do dispositivo e o dispositivo
 - Normalmente possui nível baixo
 - SO inicializa a controladora com parâmetros necessários
 - Controladora interage com o dispositivo através da interface utilizada

Hardware de E/S

- E/S mapeada na memória
 - Controlador possui registradores para comunicação com CPU
 - Por meio destes, SO pode comandar o dispositivo para entregar ou receber dados, ligar/desligar, executar tarefa
 - A partir da leitura desses registradores, SO pode descobrir o estado do dispositivo
 - Se pode receber um novo comando ou não, por exemplo.
 - Também, é comum que tenham buffer de dados
 - Que o SO pode ler ou escrever
 - Exemplo:
 - Pixels em uma tela, provenientes de uma memória RAM para vídeo
 - Neste caso, esta memória é um buffer manipulado pelos programas e pelo SO

Hardware de E/S

- E/S mapeada na memória
 - Como a CPU comunica-se com os registradores dos controladores e com os buffers de dados dos dispositivos?
 - Possibilidade 1:
 - Cada registrador de controle é associado a um número de porta de E/S (inteiro de 8 a 16 bits)
 - Conjunto de todas as portas de E/S formam o espaço de portas E/S
 - Somente o SO pode fazer acessos
 - Programas não possuem permissão para realizar acesso

Hardware de E/S

- E/S mapeada na memória
 - Como a CPU comunica-se com os registradores dos controladores e com os buffers de dados dos dispositivos?
 - Possibilidade 1:
 - Exemplo:
 - IN REG, PORT
 - colocar em REG o conteúdo de PORT
 - OUT PORT, REG
 - colocar o conteúdo de REG no registrador PORT

Hardware de E/S

- E/S mapeada na memória
 - Como a CPU comunica-se com os registradores dos controladores e com os buffers de dados dos dispositivos?
 - Possibilidade 1:
 - Maioria dos primeiros computadores utilizavam este esquema
 - Observação:
 - Espaço de endereçamento para a memória e E/S são diferentes
 - IN R0, 4 e MOV R0, 4
 - Primeira lê a porta 4 e grava em R0
 - A segunda lê a palavra de memória 4 e grava em R0

Hardware de E/S

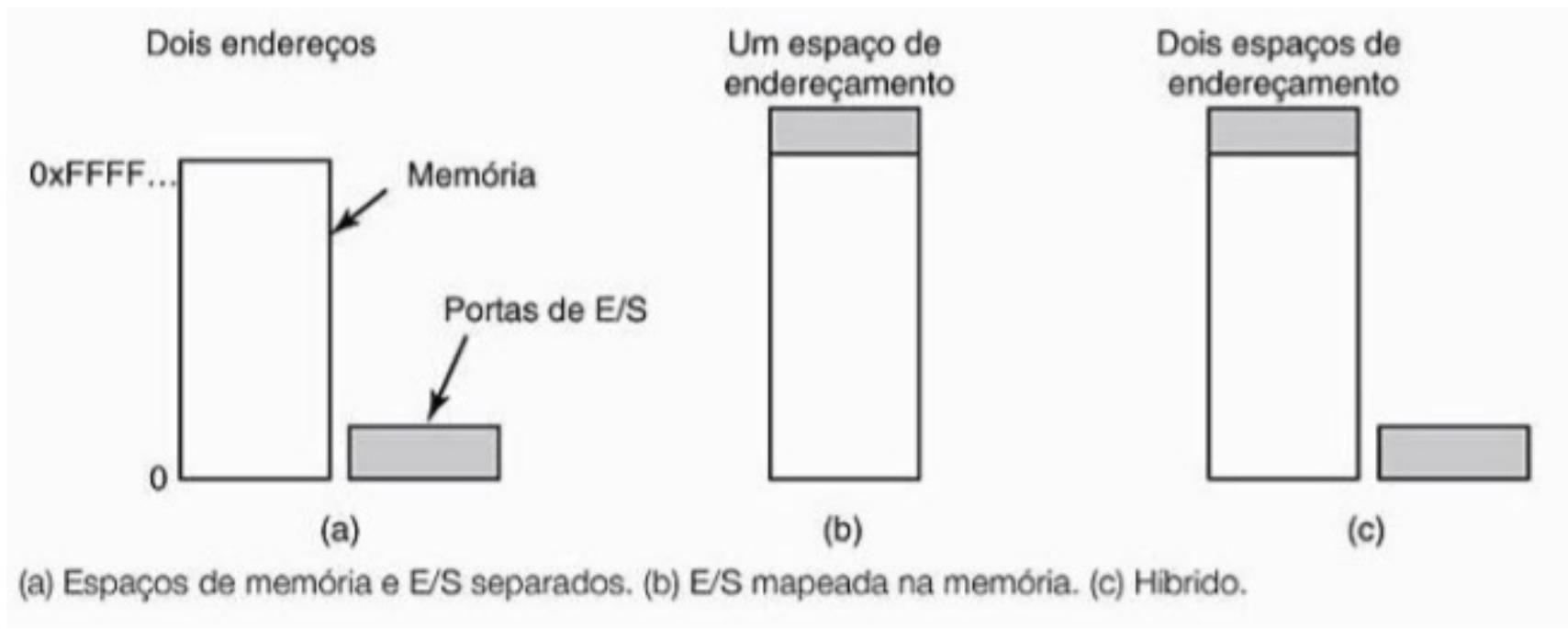
- E/S mapeada na memória
 - Como a CPU comunica-se com os registradores dos controladores e com os buffers de dados dos dispositivos?
 - Possibilidade 2:
 - Mapear registradores de controle no espaço de endereçamento da memória
 - Cada registrador de controle é associado a um endereço de memória único
 - Normalmente, estão no topo do espaço de endereçamento
 - Esquema chamado de E/S mapeada na memória

Hardware de E/S

- E/S mapeada na memória
 - Como a CPU comunica-se com os registradores dos controladores e com os buffers de dados dos dispositivos?
 - Possibilidade 3:
 - Híbrido
 - Associa os dois esquemas anteriores
 - Buffers de E/S mapeados na memória e portas de E/S separadas para os registradores de controle

Hardware de E/S

- E/S mapeada na memória
 - Como a CPU comunica-se com os registradores dos controladores e com os buffers de dados dos dispositivos?



Hardware de E/S

- Vantagens E/S mapeada na memória
 - Quando existe espaço de endereçamento separado, operações para manipulação dos registradores exige instruções específicas em baixo nível
 - Com E/S mapeada na memória isto não ocorre:
 - Registradores de controle são associadas a variáveis de memória
 - Driver de dispositivo de E/S pode ser escrito totalmente em C
 - Sem necessidade de utilizar primitivas específicas em assembly

Hardware de E/S

- Vantagens E/S mapeada na memória
 - SO não precisa proteger para que aplicativos do usuário executem E/S
 - Basta que a área de memória não seja endereçável dentro do espaço de endereçamento virtual do usuário
 - Deixa de fazer o mapeamento

Hardware de E/S

- Vantagens E/S mapeada na memória
 - Possibilidade de testar estado de registrador de controle, diretamente através de um endereço de memória
 - Por exemplo, para verificar se o dispositivo está pronto para operar

```
LOOP: TEST PORT_4 // verifica se a porta 4 é 0
      BEQ READY // se for 0, salta para
      BRANCH LOOP // caso contrário, continua
                     testando
READY:
```

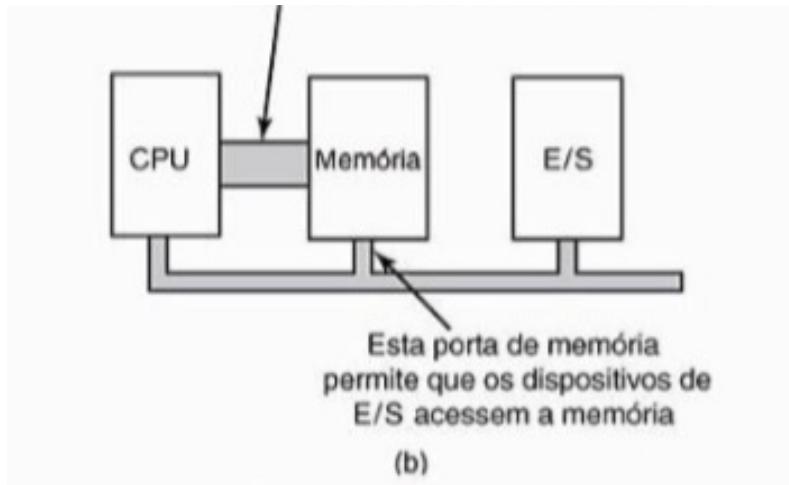
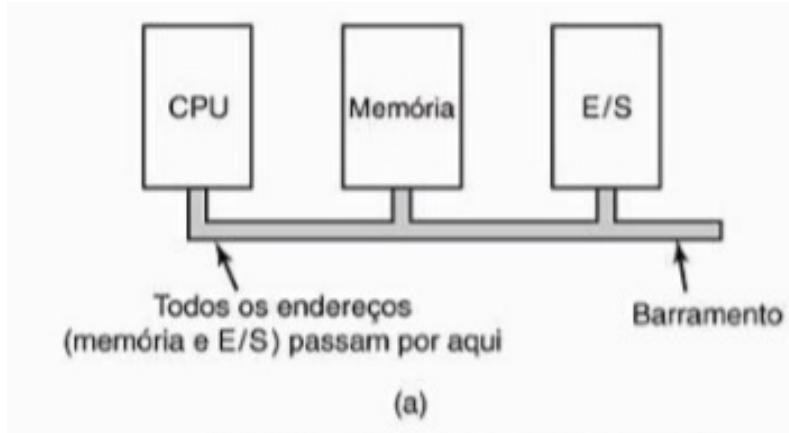
Hardware de E/S

- Desvantagens da E/S mapeada na memória
 - Uso de cache para palavras de memória
 - Uso de cache para registradores de controle não é apropriado
 - Exemplo:
 - Ao verificar repetidas vezes se o dispositivo está ocioso, após obter a primeira referência, as demais não seriam consultadas ao dispositivo
 - Solução?
 - Desabilitar seletivamente a cache para determinadas páginas de memória = complexidade

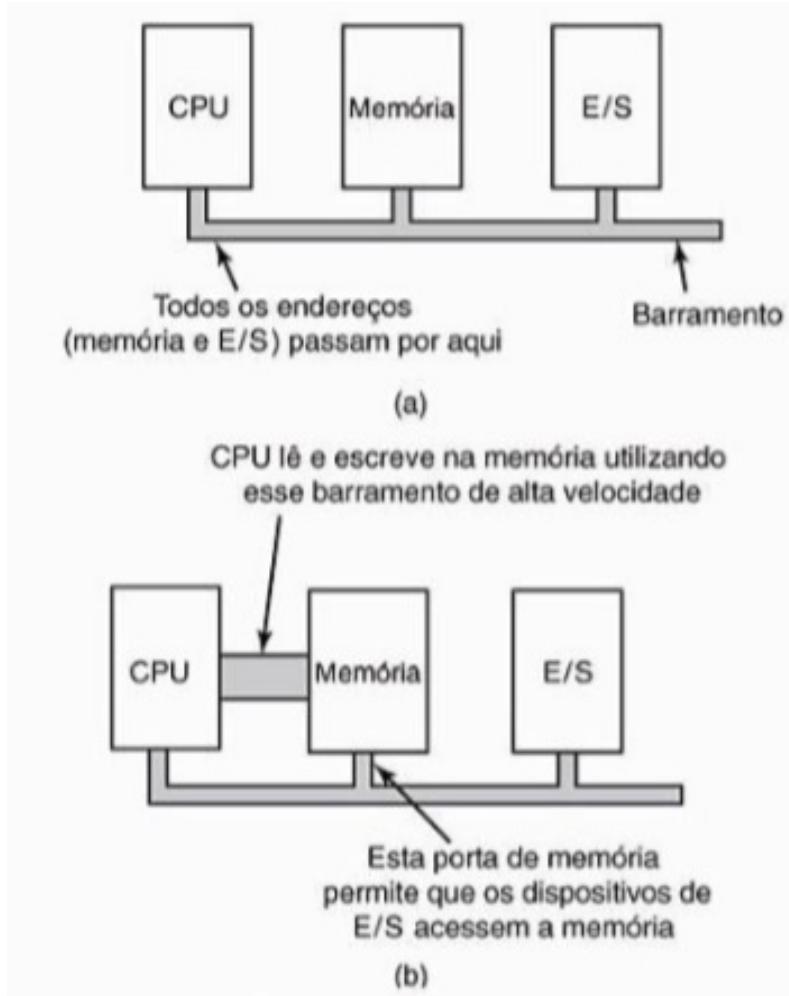
Hardware de E/S

- Desvantagens da E/S mapeada na memória
 - Enquanto existe somente um único barramento, cada componente (E/S e memória) pode responder adequadamente de acordo com o que for solicitado pela CPU
 - Tendência é ter um barramento exclusivo entre CPU e memória
 - Quando a CPU fizer referência a um endereço de memória (pelo barramento exclusivo) como E/S terá acesso para responder, se a referência for de responsabilidade dela?

Hardware de E/S



Hardware de E/S



Hardware de E/S

- Desvantagens da E/S mapeada na memória
 - Soluções:
 - 1) CPU testa memória, se der erro envia para E/S
 - 2) Memória repassa para E/S endereços de seu interesse
 - 3) Ao inicializar SO, o mesmo separa endereços que são da memória e endereços que são de E/S

Acesso direto à memória (DMA)

- Independente de ter E/S mapeada na memória ou não:
 - CPU precisa endereçar os controladores dos dispositivos
 - Para posterior troca de dados
- CPU pode requisitar, byte a byte, informações da controladora quando necessário
 - Esse esquema é bom?
 - Desperdício de tempo de CPU
 - CPU ocupada “conversando” com controladora

Acesso direto à memória (DMA)

- Acesso direto à memória (Direct Memory Access - DMA)
 - SO sempre pode usar DMA?
 - Desde que o hardware possua o controlador de DMA
 - Maioria dos sistemas possui
 - DMA possui acesso ao barramento do sistema, independente de CPU
 - Possui registradores que podem ser lidos e escritos pela CPU
 - Registrador de endereçamento de memória
 - Registrador contador de bytes
 - Registrador de controle

Acesso direto à memória (DMA)

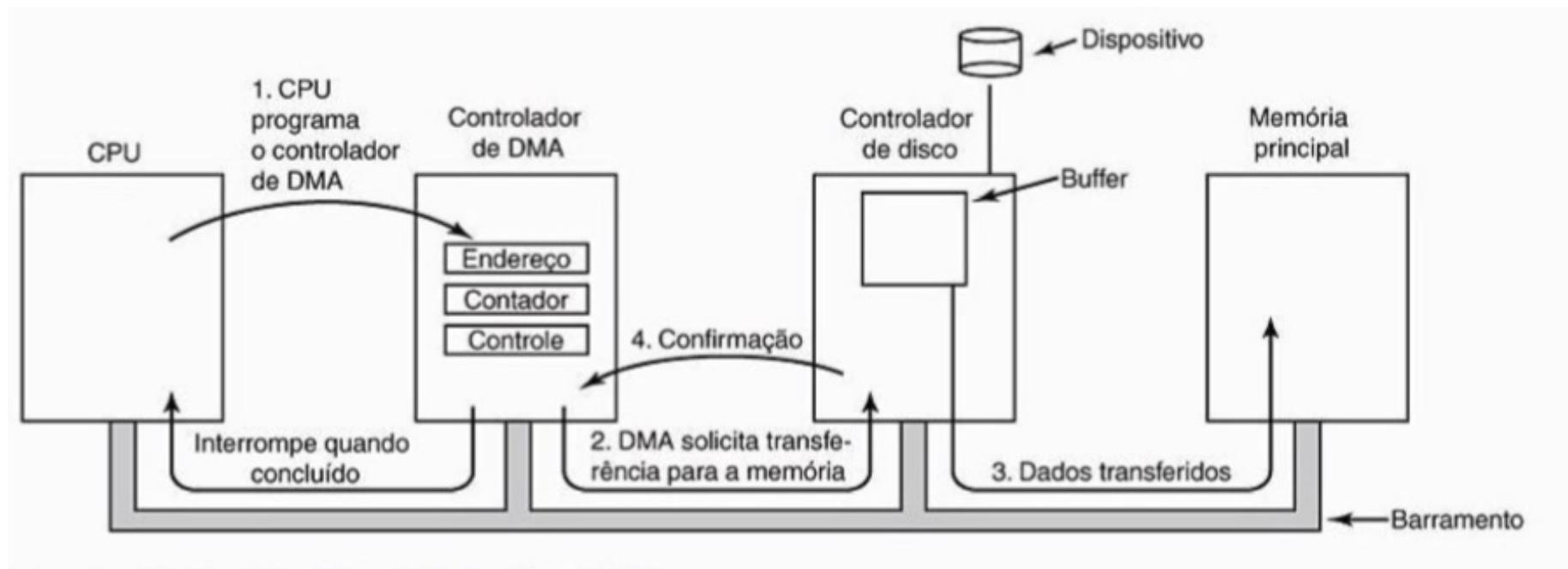
- Leitura no disco sem DMA:
 - Controlador lê um bloco do disco de maneira serial (bit a bit)
 - Controlador guarda bits até que bloco esteja pronto
 - Controlador faz soma de verificação para verificar se bloco contém erro
 - Controlador gera interrupção
 - SO faz atendimento, CPU lê o bloco do buffer do controlador
 - Leitura feita em bytes ou words (palavras)
 - Laço de repetição para leitura de toda informação do bloco
 - CPU armazena informação na memória principal

Acesso direto à memória (DMA)

- Leitura no disco **COM** DMA:
 - CPU programa controlador DMA
 - O que transferir?
 - Para onde transferir?
 - CPU faz com que controlador de disco inicie o carregamento dos dados para buffer interno (criação de blocos e verificação de erros)
 - Quando controlador de disco terminar, DMA pode iniciar!
 - DMA faz a leitura dos dados da controladora
 - Desempenha o papel que seria da CPU
 - Quando transferência está completa, DMA avisa CPU

Acesso direto à memória (DMA)

-

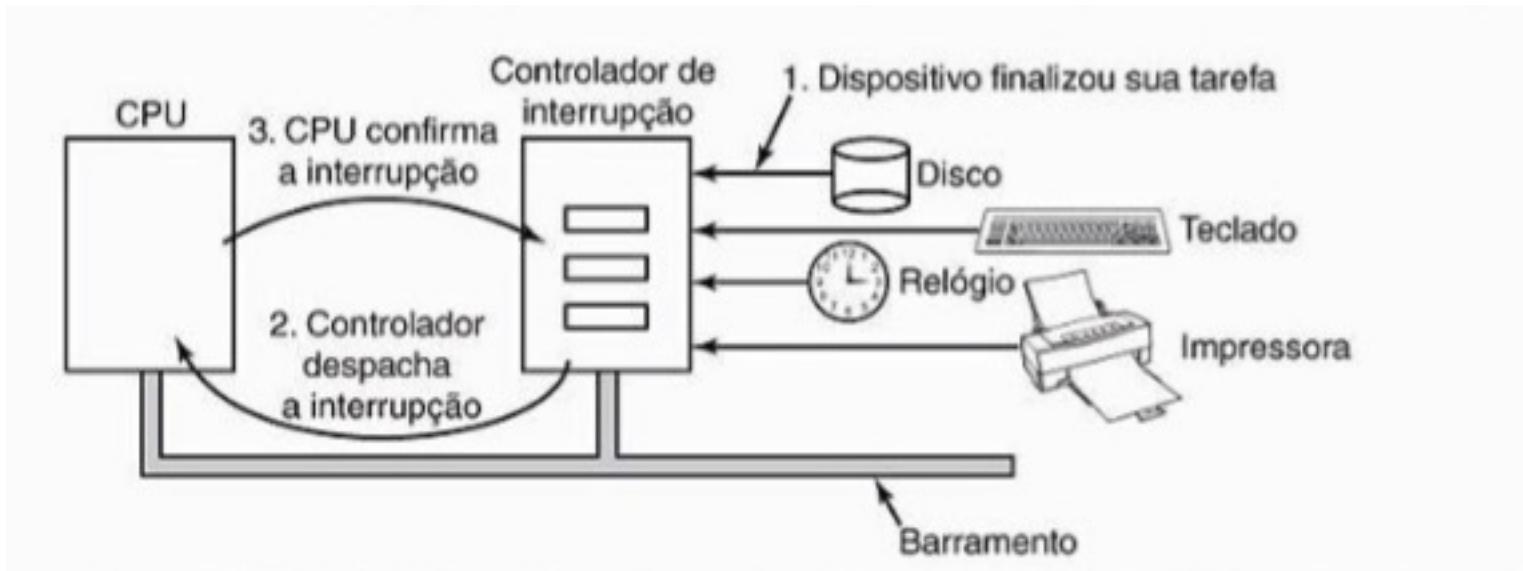


Interrupções

- Interrupções em hardware:
 - Quando dispositivo de E/S termina sua tarefa, gera uma interrupção
 - Interrupção detectada pelo controlador de interrupção
 - Se nenhuma outra interrupção está pendente, controlador de interrupção processa imediatamente
 - Caso contrário, ignora momentaneamente, embora o dispositivo continue a gerar o sinal de interrupção (até que a CPU o atenda)

Interrupções

-



Princípios do Software de E/S

- Objetivos do software de E/S
 - Independência do dispositivo
 - Deve ser possível escrever programas capazes de acessar qualquer dispositivo de E/S sem a necessidade de especificar o dispositivo
 - Ex:
 - Programa que lê um arquivo
 - Precisa se preocupar se está lendo do disco, do pendrive, do CD?
 - Nomeação uniforme
 - Cadeia de caracteres que identifica o dispositivo, independente de sua natureza

Princípios do Software de E/S

- Objetivos do software de E/S
 - Tratamento de erros
 - Deve ser realizado o mais próximo possível do hardware
 - Somente se as camadas mais baixas não forem capazes, as camadas mais altas devem tentar resolver
 - Ex:
 - Driver do dispositivo lê novamente bloco
 - Transferência síncrona ou assíncrona
 - Síncrona = bloqueante
 - Assíncrona =não bloqueante - orientada à interrupção

Princípios do Software de E/S

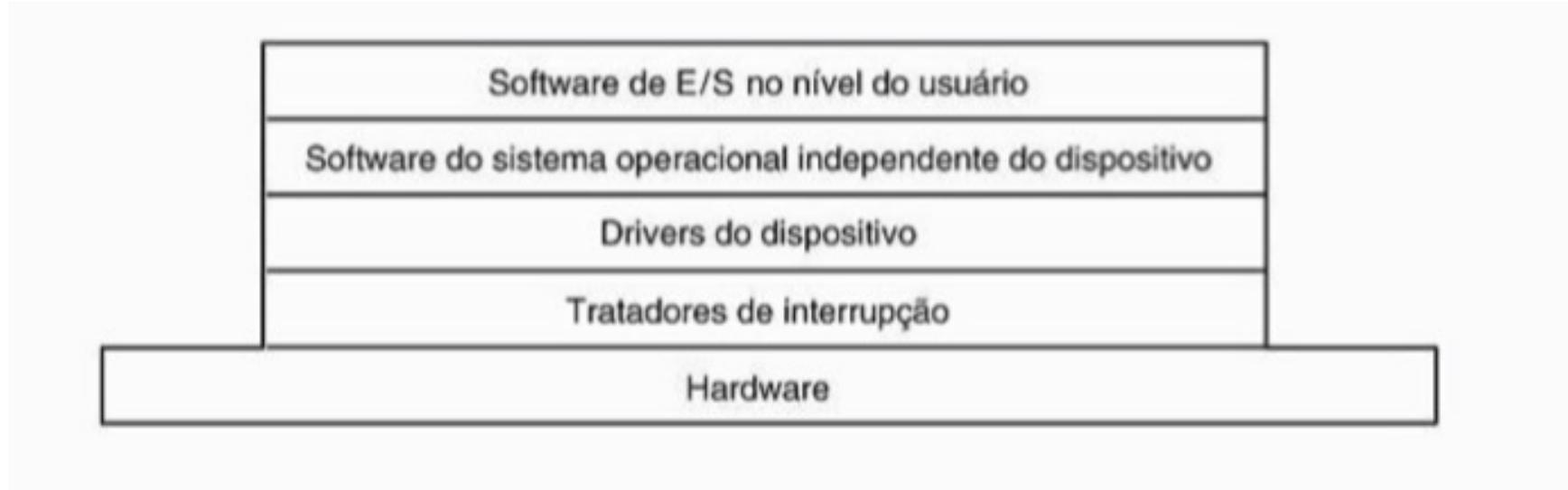
- 1) E/S Programada
 - Simples
 - Desvantagem de segurar a CPU até que toda a operação de E/S seja completada
- 2) E/S usando interrupção
 - CPU determina operação no dispositivo de E/S
 - Entre uma operação e outra (leitura ou escrita), CPU faz chaveamento de contexto para tratar de outro processo na fila de escalonamento
 - Quando o dispositivo de E/S estiver terminado a tarefa, emite uma interrupção

Princípios do Software de E/S

- 3) E/S usando DMA
 - Interrupções são caras
 - Ideia é que toda a transferência dos dados entre dispositivo de E/S e memória seja feita através de DMA
 - CPU fica livre para outras tarefas

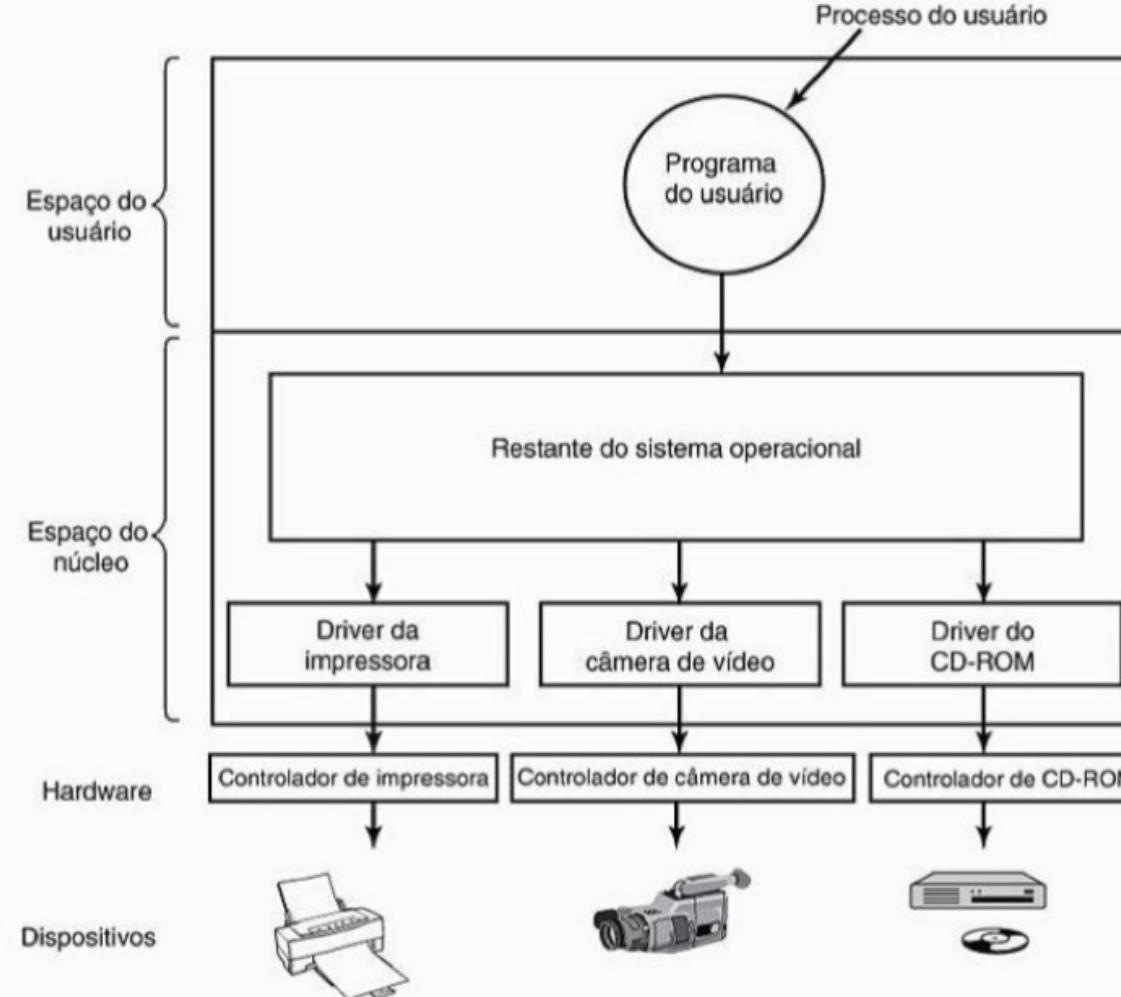
Camadas de Software de E/S

-



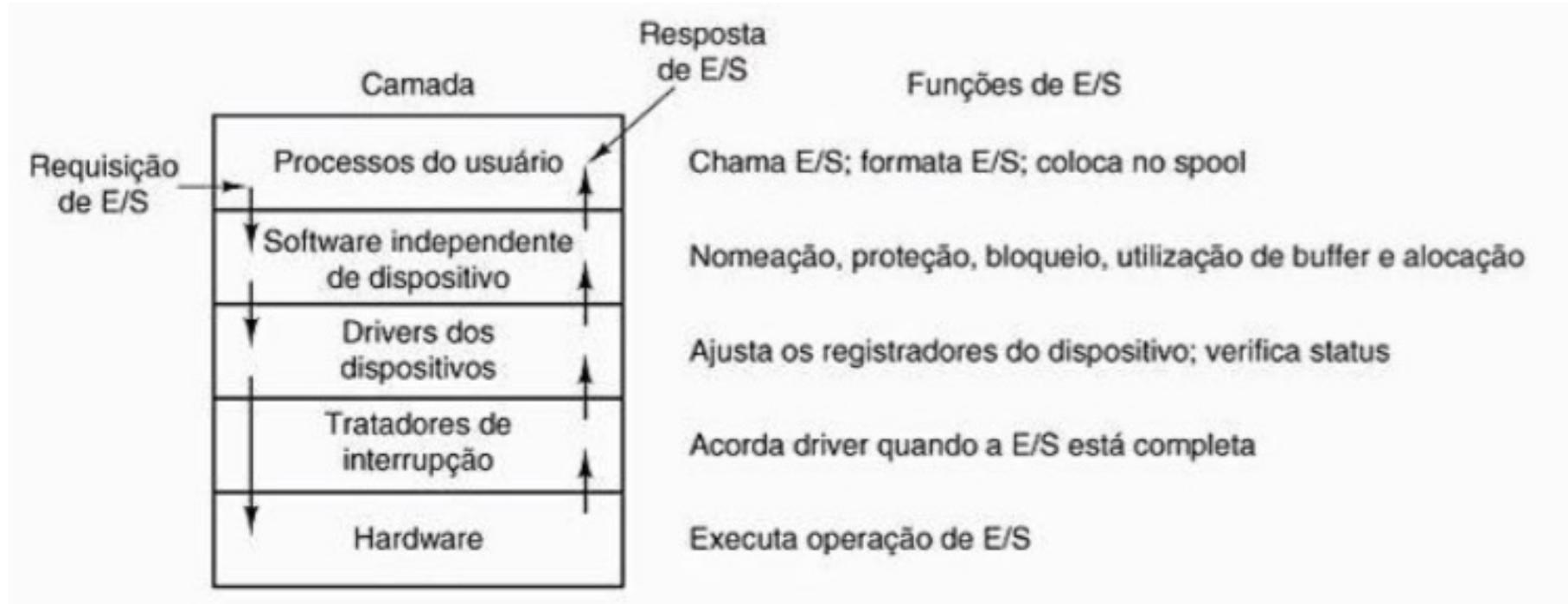
Drivers dos dispositivos

-



Camadas de Software de E/S

-



Referências

- TANENBAUM, A. S; BOS, Herbert. *Sistemas Operacionais Modernos*. São Paulo: Pearson Education do Brasil, 2016.