Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Departamento de Sistemas de Computação

SSC 140 - SISTEMAS OPERACIONAIS I
Turmas A e B

Aula 20 – Sistemas de Arquivos

Profa. Sarita Mazzini Bruschi

Slides de autoria de
Luciana A. F. Martimiano baseados no livro
Sistemas Operacionais Modernos de A. Tanenbaum

Sistema de Arquivos

- Parte do Sistema Operacional mais visível ao usuário
- Os arquivos de um sistema computacional são manipulados por meio de chamadas (*system calls*) ao Sistema Operacional;

Sistema de Arquivos

- Três importantes requisitos são considerados no armazenamento de informações:
  - Possibilidade de armazenar e recuperar uma grande quantidade de informação;
  - Informação gerada por um processo deve continuar a existir após a finalização desse processo:
    - Ex.: banco de dados;
  - Múltiplos processos podem acessar informações de forma concorrente:
    - Informações podem ser independentes de processos;

Sistema de Arquivos

- Para atender a esses requisitos, informações são armazenadas em discos (ou alguma outra mídia de armazenamento) em unidades chamadas arquivos;
- Processos podem ler ou escrever em arquivos, ou ainda criar novos arquivos;
- Informações armazenadas em arquivos devem ser persistentes, ou seja, não podem ser afetadas pela criação ou finalização de um processo;

Sistema de Arquivos

- Arquivos são manipulados pelo Sistema Operacional;
- Tarefas:
  - Estrutura de arquivos;
  - Nomes;
  - Acessos (uso);
  - Proteção;
  - Implementação;
- **SISTEMA de ARQUIVOS**: parte do SO responsável por manipular arquivos!!!

Sistema de Arquivos

- Usuário: Alto nível
  - Interface → como os arquivos aparecem;
  - Como arquivos são nomeados e protegidos;
  - Quais operações podem ser realizadas;
- SO: Baixo nível
  - Como arquivos são armazenados fisicamente;
  - Como arquivos são referenciados (*links*);

Sistema de Arquivos
Arquivos

- Arquivos:
  - Nomes;
  - Estrutura;
  - Tipos;
  - Acessos;
  - Atributos;
  - Operações;

7

Sistema de Arquivos
Nomes de arquivos

- Quando arquivos são criados, nomes são atribuídos a esses arquivos, os quais passam a são referenciados por meio desses nomes;
- Tamanho: até 255 caracteres;
  - Restrição: MS-DOS aceita de 1-8 caracteres;
- Letras, números, caracteres especiais podem compor nomes de arquivos:
  - Caracteres permitidos: A-Z, a-z, 0-9, $, %, ´, @, {, }, ~, `, !, #, (,), &
  - Caracteres **não** permitidos: ?, *, /, \, “, |, <, >, :

8

Sistema de Arquivos
Nomes de arquivos

- Alguns Sistemas Operacionais são sensíveis a letras maiúsculas e minúsculas *(case sensitive)* e outros não;
  - UNIX é sensível :
    - Ex.: exemplo.c é diferente de Exemplo.c;
  - MS-DOS não é sensível:
    - Ex.: exemplo.c é o mesmo que Exemplo.c;
- Win95/Win98/WinNT/Win2000/WinXP/WinVista herdaram características do sistema de arquivos do MS-DOS;
  - No entanto, WinNT/Win2000/WinXP/WinVista possuem um sistema de arquivos próprio → NTFS (*New Technology File System*);

9

Sistema de Arquivos
Nomes de arquivos

- Alguns sistemas suportam uma extensão relacionada ao nome do arquivo:
  - MS-DOS: 1-3 caracteres; suporta apenas uma extensão;
  - UNIX:
    - Extensão pode conter mais de 3 caracteres;
    - Suporta mais de uma extensão: Ex.: exemplo.c.Z (arquivo com compressão);
    - Permite que arquivos sejam criados sem extensão;

10

Sistema de Arquivos
Nomes de arquivos

- Uma extensão, geralmente, associa o arquivo a algum aplicativo (associação feita pelo aplicativo):
  - .doc – Microsoft Word;
  - .c – Compilador C;
- SO pode ou não associar as extensões aos aplicativos:
  - Unix não associa;
  - Windows associa;

11

Sistema de Arquivos
Estrutura de arquivos

- Arquivos podem ser estruturados de diferentes maneiras:
  - a) Seqüência não estruturada de bytes
    - Para o SO arquivos são apenas conjuntos de bytes;
    - SO não se importa com o conteúdo do arquivo;
      - Significado deve ser atribuído pelos programas em nível de usuário (aplicativos);
    - Vantagem:
      - Flexibilidade: os usuários nomeiam seus arquivos como quiserem;
    - Ex.: UNIX e Windows;

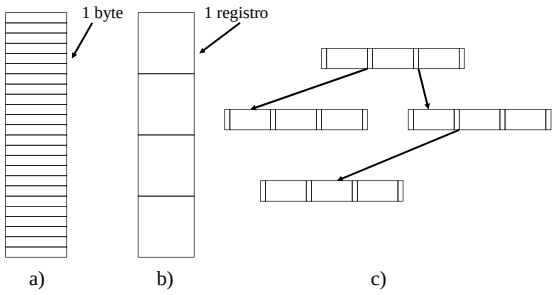
12

Sistema de Arquivos
Estrutura de arquivos

- b) Seqüência de registros de tamanho fixo, cada qual com uma estrutura interna → leitura/escrita são realizadas em registros;
  - SOs mais antigos → *mainframes* e cartões perfurados (80 caracteres);
  - Nenhum sistema atual utiliza esse esquema;
- c) Árvores de registros (tamanho variado), cada qual com um campo **chave** em uma posição fixa:
  - SO decide onde colocar os arquivos;
  - Usado em *mainframes* atuais;

13

Sistema de Arquivos
Estrutura de arquivos



14

Sistema de Arquivos
Tipos de arquivos

- **Arquivos regulares** → são aqueles que contêm informações dos usuários;
- **Diretórios** → são arquivos responsáveis por manter a estrutura do Sistema de Arquivos;
- **Arquivos especiais de caracteres** → são aqueles relacionados com E/S e utilizados para modelar dispositivos seriais de E/S;
  - Ex.: impressora, interface de rede, terminais;
- **Arquivos especiais de bloco** → são aqueles utilizados para modelar discos;

15

Sistema de Arquivos
Tipos de arquivos

- Arquivos regulares podem ser de dois tipos:
  - ASCII:
    - Consistem de linhas de texto;
    - Facilitam integração de arquivos;
    - Podem ser exibidos e impressos como são;
    - Podem ser editados em qualquer Editor de Texto;
    - Ex.: arquivos texto;
  - Binário:
    - Todo arquivo não ASCII;
    - Possuem uma estrutura interna conhecida pelos aplicativos que os usam;
    - Ex.: programa executável;

16

Sistema de Arquivos
Acessos em arquivos

- SOs mais antigos ofereciam apenas acesso seqüencial no disco → leitura em ordem byte a byte (registro a registro);
- SOs mais modernos fazem acesso randômico ou aleatório;
  - Acesso feito por **chave**;
    - Ex.: base de dados de uma empresa de aérea;
  - Métodos para especificar onde iniciar leitura:
    - Operação Read → posição do arquivo em que se inicia a leitura;
    - Operação Seek → marca posição corrente permitindo leitura seqüencial;

17

Sistema de Arquivos
Atributos de arquivos

- Além do nome e dos dados, todo arquivo tem outras informações associadas a ele → **atributos**;
- A lista de atributos varia de SO para SO;

18

Sistema de Arquivos
Atributos de arquivos

Atributo	Significado
Proteção	Quem acesso o arquivo e de que maneira
Senha	Chave para acesso ao arquivo
Criador	Identificador da pessoa que criou o arquivo
Dono	Dono corrente
Flag de leitura	0 para leitura/escrita; 1 somente para leitura
Flag de oculto	0 para normal; 1 para não aparecer
Flag de sistema	0 para arquivos normais; 1 para arquivos do sistema
Flag de repositório	0 para arquivos com *backup*; 1 para arquivos sem *backup*

19

Sistema de Arquivos
Atributos de arquivos

Atributo	Significado
Flag ASCII/Binary	0 para arquivo ASCII; 1 para arquivo binário
Flag de acesso aleatório	0 para arquivo de acesso seqüencial; 1 para arquivo de acesso randômico
Flag de temporário	0 para normal; 1 para temporário
Flag de impedido	0 para arquivo desimpedido; diferente de 0 para arquivo impedido
Tamanho do registro	Número de bytes em um registro
Posição da chave	Deslocamento da chave em cada registro
Tamanho da chave	Número de bytes no campo chave (*key*)

20

Sistema de Arquivos
Atributos de arquivos

Atributo	Significado
Momento da criação	Data e hora que o arquivo foi criado
Momento do último acesso	Data e hora do último acesso ao arquivo
Momento da última mudança	Data e hora da última modificação do arquivo
Tamanho	Número de bytes do arquivo
Tamanho Máximo	Número máximo de bytes que o arquivo pode ter

21

Sistema de Arquivos
Operações em arquivos

- Diferentes sistemas provêm diferentes operações que permitem armazenar e recuperar arquivos;
- Operações mais comuns (*system calls*):
  - `Create`; `Delete`;
  - `Open`; `Close`;
  - `Read`; `Write`; `Append`;
  - `Seek`;
  - `Get attributes`; `Set attributes`;
  - `Rename`;

22

Sistema de Arquivos
Arquivos mapeados em memória

- Alguns SOs permitem que arquivos sejam mapeados diretamente no espaço de endereçamento (virtual) de um processo em execução → acesso mais rápido;
- *System Calls*: `Map` e `unmap`;
- Funciona melhor em sistemas que suportam segmentação;

23

Sistema de Arquivos
Arquivos mapeados em memória

- Problemas:
  - Difícil prever o tamanho de arquivos de saída;
  - Compartilhamento de arquivos entre diferentes processos → SO não deve permitir acesso a arquivos com dados inconsistentes;
  - Arquivo pode ser maior que um segmento ou maior que o espaço virtual utilizado → mapear pequenas partes do arquivo;

24

Sistema de Arquivos
Diretórios

- **Diretórios** → são arquivos responsáveis por manter a estrutura do Sistema de Arquivos;
  - Organização;
  - Operações;

25

Sistema de Arquivos
Diretórios

- Organização pode ser feita das seguintes maneiras:
  - Nível único (*Single-level*);
  - Dois níveis (*Two-level*);
  - Hierárquica;

26

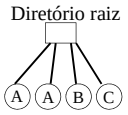
Sistema de Arquivos
Diretórios – Nível único

- Apenas um diretório contém todos os arquivos → diretório raiz (*root directory*);
- Computadores antigos utilizavam esse método, pois eram monousuários;
- Exceção: CDC 6600 → supercomputador que utilizava-se desse método, apesar de ser multiusuário;
- Vantagens:
  - Simplicidade;
  - Eficiência;

27

Sistema de Arquivos
Diretórios – Nível único

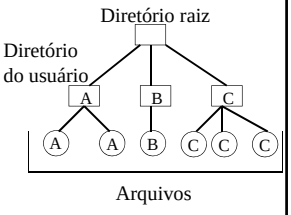
- 04 arquivos;
- Três diferentes proprietários;
- Desvantagens:
  - Sistemas multiusuários: Diferentes usuários podem criar arquivos como mesmo nome;
  - Exemplo:
    - Usuários A e B criam, respectivamente, um arquivo *mailbox*;
    - Usuário B sobrescreve arquivo do usuário A



28

Sistema de Arquivos
Diretórios – Dois níveis

- Cada usuário possui um diretório privado;
- Sem conflitos de nomes de arquivos;
- Procedimento de *login*: identificação;
- Compartilhamento de arquivos → programas executáveis do sistema;
- Desvantagem:
  - Usuário com muitos arquivos;



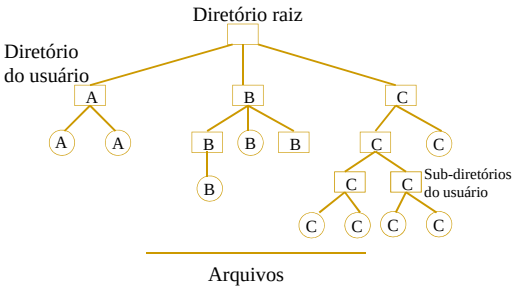
29

Sistema de Arquivos
Diretórios – Hierárquico

- Hierarquia de diretórios → árvores de diretórios;
  - Usuários podem querer agrupar seus arquivos de maneira lógica, criando diversos diretórios que agrupam arquivos;
- Sistemas operacionais modernos utilizam esse método;
- Flexibilidade;

30

Sistema de Arquivos
Diretórios – Hierárquico



31

Sistema de Arquivos
Diretórios – Caminho (*path name*)

- O método hierárquico requer métodos pelos quais os arquivos são acessados;
- Dois métodos diferentes:
  - Caminho absoluto (*absolute path name*);
  - Caminho relativo (*relative path name*);

32

Sistema de Arquivos
Diretórios – Caminho (*path name*)

- Caminho absoluto: consiste de um caminho a partir do diretório raiz até o arquivo;
  - É ÚNICO;
  - Funciona independentemente de qual seja o diretório corrente;
  - Ex.:
    - UNIX: */usr/ast/mailbox*;
    - Windows: *\usr\ast\mailbox*;

33

Sistema de Arquivos
Diretórios – Caminho (*path name*)

- Diretório de Trabalho (*working directory*) ou diretório corrente (*current directory*);
- Caminho relativo é utilizado em conjunto com o diretório corrente;
- Usuário estabelece um diretório como sendo o diretório corrente; nesse caso caminhos não iniciados no diretório raiz são tido como relativos ao diretório corrente;
  - Exemplo:
    - *cp /usr/ast/mailbox /usr/ast/mailbox.bak*
    - *Diretório corrente: /usr/ast → cp mailbox mailbox.bak*

34

Sistema de Arquivos
Diretórios – Caminho (*path name*)

- "." → diretório corrente;
- ".." → diretório pai (anterior ao corrente);
- Ex.: diretório corrente /usr/ast:

```
cp ../lib/dictionary .
cp /usr/lib/dictionary .
cp /usr/lib/dictionary  dictionary
cp /usr/lib/dictionary /usr/ast/dictionary
```

35

Sistema de Arquivos
Diretórios – Operações

- `Create`; `Delete`;
- `Opendir`; `Closedir`;
- `Readdir`;
- `Rename`;
- `Link` (um arquivo pode aparecer em mais de um diretório);
- `Unlink`;

36

Implementando o Sistema de arquivos

- Implementação do Sistema de Arquivos:
  - Como arquivos e diretórios são armazenados;
  - Como o espaço em disco é gerenciado;
  - Como tornar o sistema eficiente e confiável;

37

Implementando o Sistema de arquivos - *Layout*

- Arquivos são armazenados em discos;
- Discos podem ser divididos em uma ou mais partições, com sistemas de arquivos independentes;
- Setor 0 do disco é destinado ao MBR – *Master Boot Record*; que é responsável pela a tarefa de *boot* do computador;
  - MBR possui a tabela de partição, com o endereço inicial e final de cada partição;
  - BIOS lê e executa o MBR;

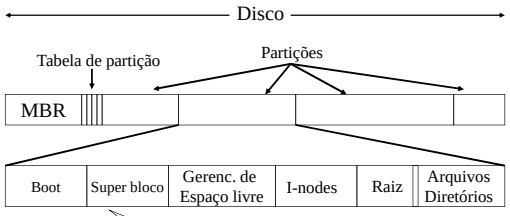
38

Implementando o Sistema de arquivos - *Layout*

- Tarefas básicas do MBR (pode variar dependendo do SO):
  - 1ª → localizar a partição ativa;
  - 2ª → ler o primeiro bloco dessa partição, chamado bloco de *boot* (*boot block*);
  - 3ª → executar o bloco de *boot* ;
- *Layout* de um Sistema de Arquivos pode variar; mas a idéia geral é a seguinte:

39

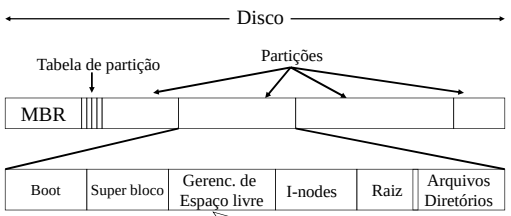
Implementando o Sistema de arquivos - *Layout*



Contém parâmetros (tipo do SA, número de blocos) sobre o sistema de arquivos e é carregado na memória

40

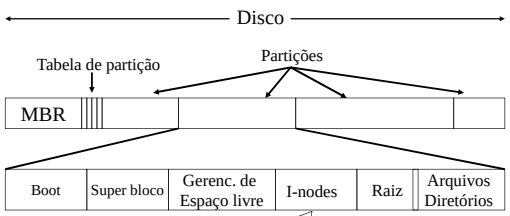
Implementando o Sistema de arquivos - *Layout*



Contém informações sobre os blocos livres do disco (mapa de bits ou lista encadeada)

41

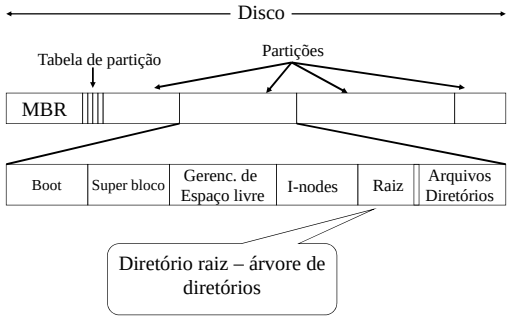
Implementando o Sistema de arquivos - *Layout*



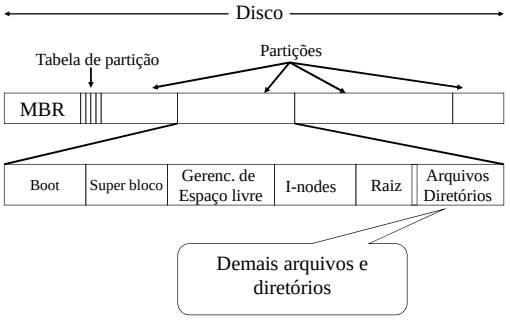
Estruturas de dados (vetor) contendo informações sobre os arquivos

42

Implementando o Sistema de arquivos - *Layout*



Implementando o Sistema de arquivos - *Layout*



Implementando o Sistema de arquivos - Arquivos

- Armazenamento de arquivos → como os arquivos são alocados no disco;
- Diferentes técnicas são implementas por diferentes Sistemas Operacionais;
  - Alocação contínua;
  - Alocação com lista encadeada;
  - Alocação com lista encadeada utilizando uma tabela na memória (FAT);
  - *I-Nodes*;

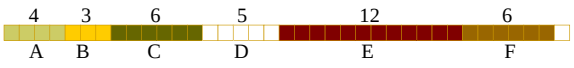
Implementando o Sistema de arquivos - Arquivos

- Alocação contínua:
  - Técnica mais simples;
  - Armazena arquivos de forma contínua no disco;
    - Ex.: em um disco com blocos de 1kb um arquivo com 50kb será alocado em 50 blocos consecutivos;

Implementando o Sistema de arquivos - Arquivos

- Alocação contínua:

37 Blocos



Removendo os arquivos D e F...

Implementando o Sistema de arquivos - Arquivos

- Alocação contínua:
  - Vantagens:
    - Simplicidade: somente o endereço do primeiro bloco e número de blocos no arquivo são necessários;
    - Desempenho para o acesso ao arquivo: acesso seqüencial;
  - Desvantagens (discos rígidos):
    - Fragmentação externa:
      - Compactação → alto custo;
      - Reuso de espaço → atualização da lista de espaços livres;
        - Conhecimento prévio do tamanho do arquivo para alocar o espaço necessário;
  - CD-ROM e DVD-ROM (quando somente escrita);

Implementando o Sistema de arquivos - Arquivos

- Alocação com lista encadeada:
  - A primeira palavra de cada bloco é um ponteiro para o bloco seguinte;
  - O restante do bloco é destinado aos dados;
  - Apenas o endereço em disco do primeiro bloco do arquivo é armazenado;
    - Serviço de diretório é responsável por manter esse endereço;

49

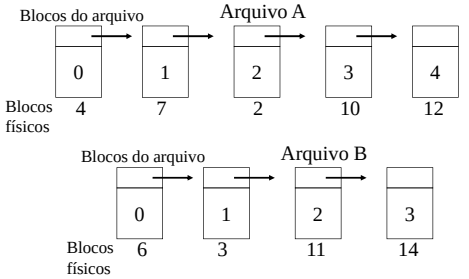
Implementando o Sistema de arquivos - Arquivos

- Alocação com lista encadeada:
  - Desvantagens:
    - Acesso aos arquivos é feito aleatoriamente → processo mais lento;
    - A informação armazenada em um bloco não é mais uma potência de dois, pois existe a necessidade de se armazenar o ponteiro para o próximo bloco;
  - Vantagem:
    - Não se perde espaço com a fragmentação externa;

50

Implementando o Sistema de arquivos - Arquivos

- Alocação com lista encadeada:



51

Implementando o Sistema de arquivos - Arquivos

- Alocação com lista encadeada utilizando uma tabela na memória:
  - O ponteiro é colocado em uma tabela na memória ao invés de ser colocado no bloco;
    - FAT → Tabela de alocação de arquivos (*File Allocation Table*);
    - Assim, todo o bloco está disponível para alocação de dados;
  - Serviço de diretório é responsável por manter o início do arquivo (bloco inicial);
  - MS-DOS e família Windows 9x (exceto WinNT, Win2000 e WinXP - NTFS);

52

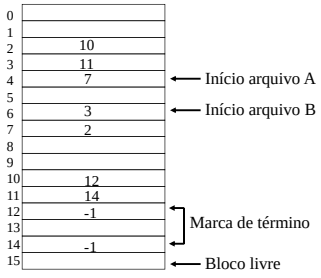
Implementando o Sistema de arquivos - Arquivos

- Acesso aleatório se torna mais fácil devido ao uso da memória;
- Desvantagem:
  - Toda a tabela deve estar na memória;
  - Exemplo:
    - Com um disco de 20Gb com blocos de 1kb, a tabela precisa de 20 milhões de entradas, cada qual com 3 bytes (para permitir um acesso mais rápido, cada entrada pode ter 4 bytes) ocupando 60 (80) Mb da memória;

53

Implementando o Sistema de arquivos - Arquivos

- Alocação com lista encadeada utilizando FAT



54

Implementando o Sistema de arquivos - Arquivos

- *I-nodes*:
  - Cada arquivo possui uma estrutura de dados chamada ***i-node*** (*index-node*) que lista os atributos e endereços em disco dos blocos do arquivo;
    - Assim, dado o *i-node* de um arquivo é possível encontrar todos os blocos desse arquivo;
  - Se cada *i-node* ocupa **n** bytes e **k** arquivos podem estar aberto ao mesmo tempo → o total de memória ocupada é **kn** bytes;
  - UNIX e Linux;

55

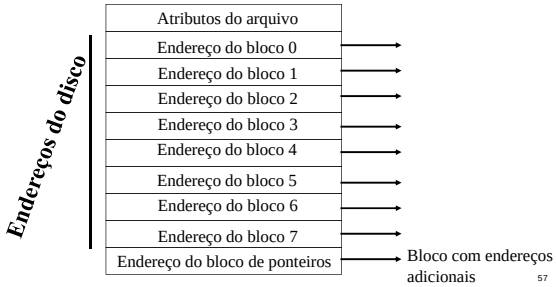
Implementando o Sistema de arquivos - Arquivos

- Espaço de memória ocupado pelos *i-nodes* é proporcional ao número de arquivos abertos; enquanto o espaço de memória ocupado pela tabela de arquivo (FAT) é proporcional ao tamanho do disco;
- Vantagem:
  - O *i-node* somente é carregado na memória quando o seu respectivo arquivo está aberto (em uso);
- Desvantagem:
  - O tamanho do arquivo pode aumentar muito
    - Solução: reservar o último endereço para outros endereços de blocos;

56

Implementando o Sistema de arquivos - Arquivos

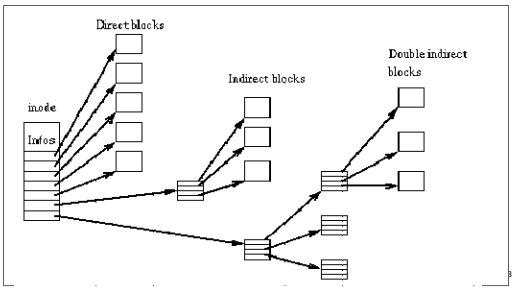
- *I-nodes*:



57

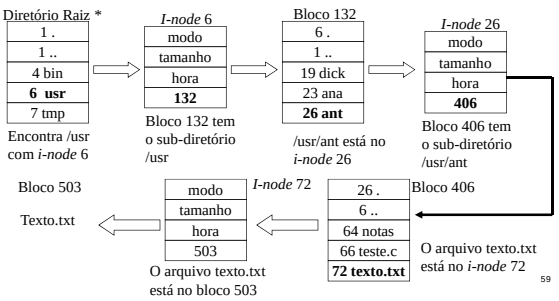
Implementando o Sistema de arquivos - Arquivos

- *I-nodes*:



Implementando o Sistema de arquivos - Arquivos

- *I-nodes*



59