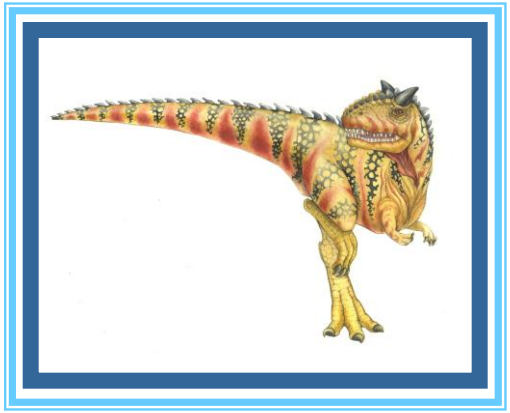
Capítulo 10: Interface de Sistemas de Arquivos



Sobre a apresentação (About the slides)

Os slides e figuras dessa apresentação foram criados por Silberschatz, Galvin e Gagne em 2009. Esse apresentação foi modificada por Cristiano Costa (cac@unisinos.br). Basicamente, os slides originais foram traduzidos para o Português do Brasil.

É possível acessar os slides originais em http://www.os-book.com

Essa versão pode ser obtida em http://www.inf.unisinos.br/~cac

The slides and figures in this presentation are copyright Silberschatz, Galvin and Gagne, 2009. This presentation has been modified by Cristiano Costa (cac@unisinos.br). Basically it was translated to Brazilian Portuguese.

You can access the original slides at http://www.os-book.com

This version could be downloaded at http://www.inf.unisinos.br/~cac

Capítulo 10: Interface de Sistemas de Arquivos

- Conceito de Arquivo
- Métodos de Acesso
- Estrutura de Diretórios
- Montagem de Sistema de Arquivos
- Compartilhamento de Arquivos
- Proteção

Objetivos

- Explicar a função de sistemas de arquivos
- Descrever a interface para sistemas de arquivos
- Discutir questões de projeto de sistemas de arquivos, incluindo métodos de acesso, compartilhamento de arquivos, travamento (*lock*) de arquivos e estruturas de diretórios
- Explorar a proteção de sistemas de arquivos

Conceito de Arquivo

- Espaço de endereçamento lógico contíguo
- Tipos:
  - Dados
    - numérico
    - caractere
    - binário
  - Programa

Estrutura de Arquivos

- Nenhuma - seqüência de palavras, bytes
- Estrutura de registro simples
  - Linhas
  - Tamanho fixo
  - Tamanho variável
- Estruturas Complexas
  - Documentos formatados
  - Arquivo de carga relocável
- Pode simular os dois últimos com o primeiro método inserindo caracteres especiais de controle.
- Quem decide:
  - Sistema Operacional
  - Programa

Atributos de Arquivos

- **Nome** – única informação mantida em uma forma legível para o usuário.
- **Identificador –** identificador único (número) do arquivo pelo sistema de arquivos
- **Tipo** – necessária para sistemas que suportam diferentes tipos de arquivos.
- **Localização** – ponteiro para a posição do arquivo no dispositivo.
- **Tamanho** – tamanho atual do arquivo.
- **Proteção** – controla quem pode ler, escrever e executar.
- **Hora**, **data**, **e identificação do usuário** – dados para proteção, segurança e monitoração de uso.
- Informações sobre os arquivos são mantidas nas estruturas de diretórios, as quais são armazenadas no disco.

Operações sobre Arquivos

- Arquivo é um **tipo de dados abstrato**
- **Criar** (*create*)
- **Escrever** (*write*)
- **Ler** (*read*)
- **Reposicionamento de um arquivo** (*seek*)
- **Excluir** (*delete*)
- **Truncamento** (*truncate*)
- **Abrir**[F_i] (*open*[F_i]) – procura na estrutura de diretório do disco pela entrada F_i, e move o conteúdo da entrada para a memória.
- **Fechar**[F_i] (*close*[F_i]) – move o conteúdo da entrada F_i na memória para a estrutura de diretório no disco.

Arquivos Abertos

- Alguns dados são necessários para gerenciar arquivos abertos:
  - Ponteiro de Arquivo: ponteiro para a última localização de leitura/escrita, por processo que tem um arquivo aberto
  - Contador de arquivos abertos: contador do número de vezes que um arquivo é aberto - para permitir a remoção dos dados da tabela de arquivos abertos quando o último processo fechar o arquivo
  - Localização no disco do arquivo: cache dos dados acessados
  - Direitos de acesso: informações de modo de acesso por processo

Travamento (*Lock*) de Arquivos Abertos

- Fornecido por alguns sistemas operacionais e sistemas de arquivos
- Media o acesso a um arquivo
- Mandatário ou consultivo:
  - **Mandatário** – acesso é negado dependendo das travas mantidas e requeridas
  - **Consultivo** – processos podem buscar por status das travas e decidir o que fazer

Exemplo de Travamento de Arquivo – Java API

```
import java.io.*;
import java.nio.channels.*;
public class LockingExample {
    public static final boolean EXCLUSIVE = false;
    public static final boolean SHARED = true;
    public static void main(String arsg[]) throws IOException {
        FileLock sharedLock = null;
        FileLock exclusiveLock = null;
        try {
            RandomAccessFile raf = new RandomAccessFile("file.txt", "rw");
            // get the channel for the file
            FileChannel ch = raf.getChannel();
            // this locks the first half of the file - exclusive
            exclusiveLock = ch.lock(0, raf.length()/2, EXCLUSIVE);
            /** Now modify the data . . . */
            // release the lock
            exclusiveLock.release();
```

Exemplo de Travamento de Arquivo – Java API

```
                    // this locks the second half of the file - shared
                    sharedLock = ch.lock(raf.length()/2+1, raf.length(),
                                         SHARED);
                    /** Now read the data . . . */
                    // release the lock
                    exclusiveLock.release();
            } catch (java.io.IOException ioe) {
                    System.err.println(ioe);
            }finally {
                    if (exclusiveLock != null)
                    exclusiveLock.release();
                    if (sharedLock != null)
                    sharedLock.release();
            }
    }
}
```

Tipos de Arquivos – Nome, Extensão

file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm, mp3, avi	binary file containing audio or A/V information

Métodos de Acesso

- **Acesso Sequencial**

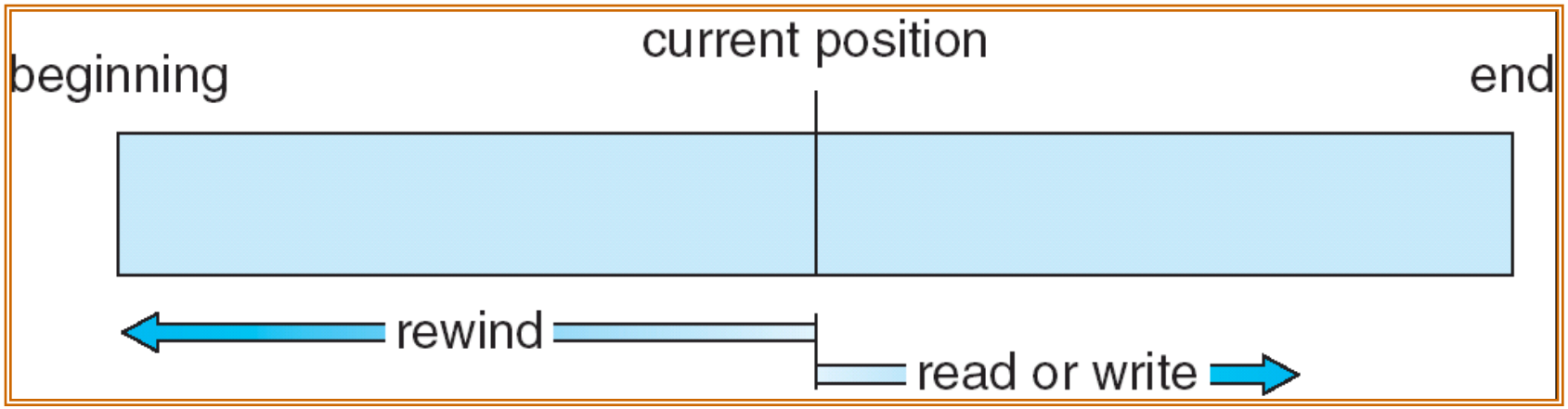
  read next
  write next
  reset
  no read after last write
  (rewrite)

- **Acesso Direto**

  read *n*
  write *n*
  position to *n*
  read next
  write next
  rewrite *n*

  n = número do bloco relativo

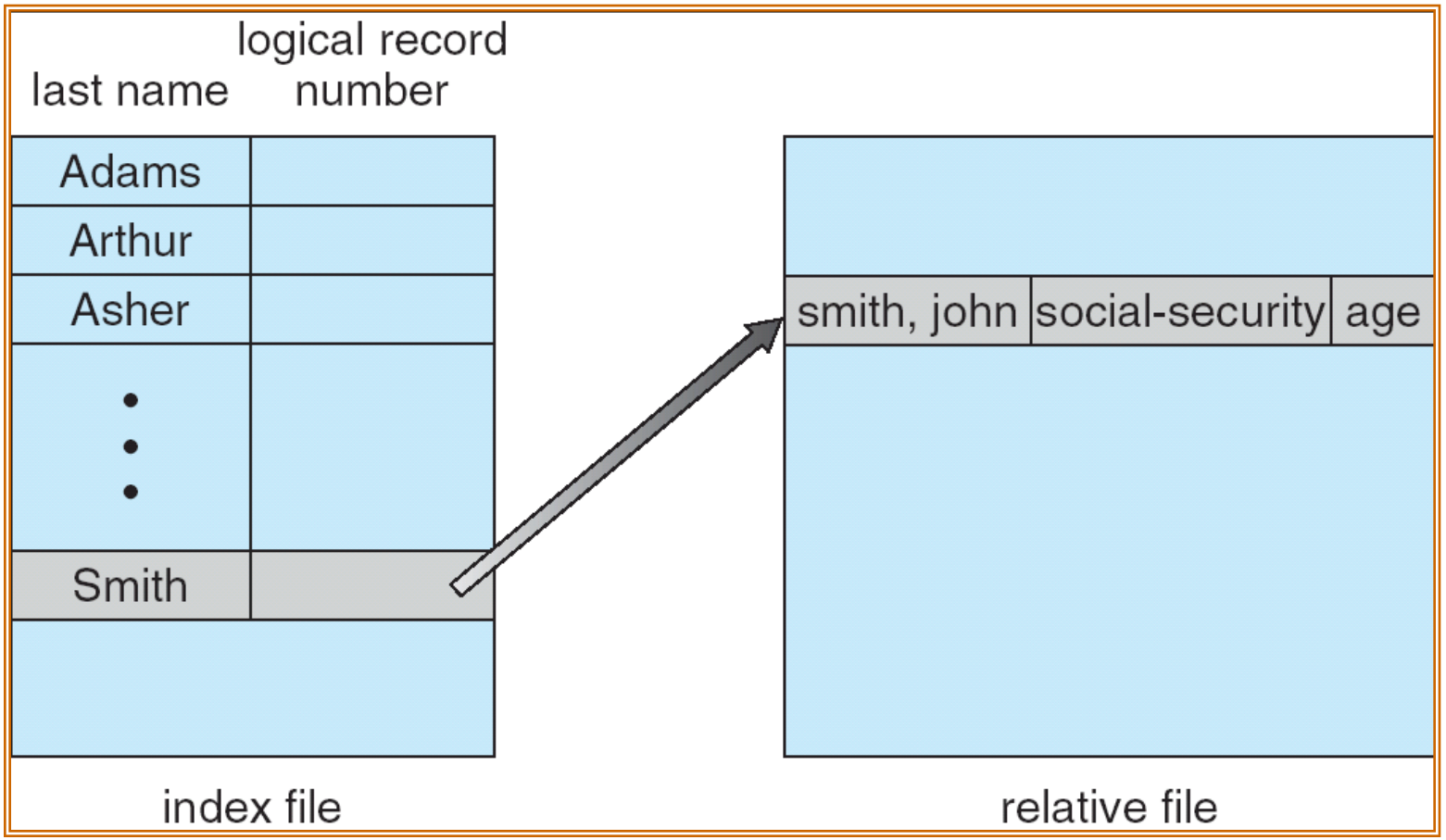
Acesso Sequencial a Arquivos



Simulação de Acesso Sequencial em Acesso Direto

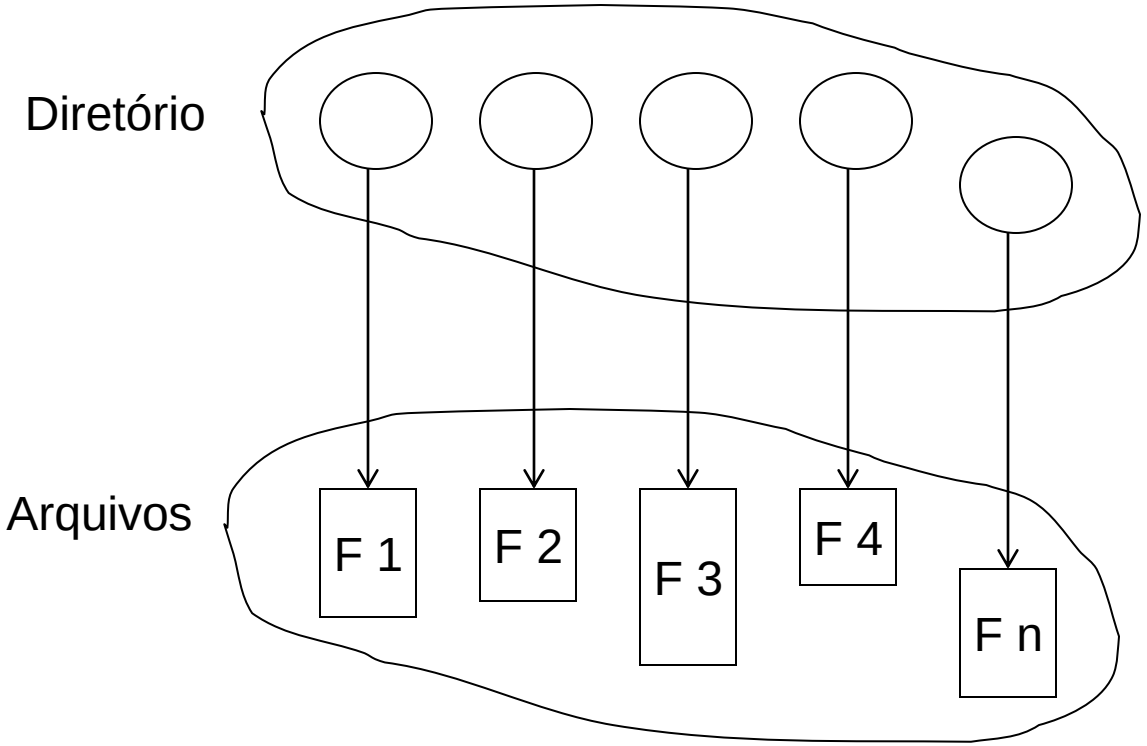
sequential access	implementation for direct access
reset	$cp = 0;$
read next	*read cp*; $cp = cp + 1;$
write next	*write cp*; $cp = cp + 1;$

Exemplo de Índice e Arquivos Relativos



Estrutura de Diretório

- Uma coleção de nodos contendo informações sobre todos arquivos.

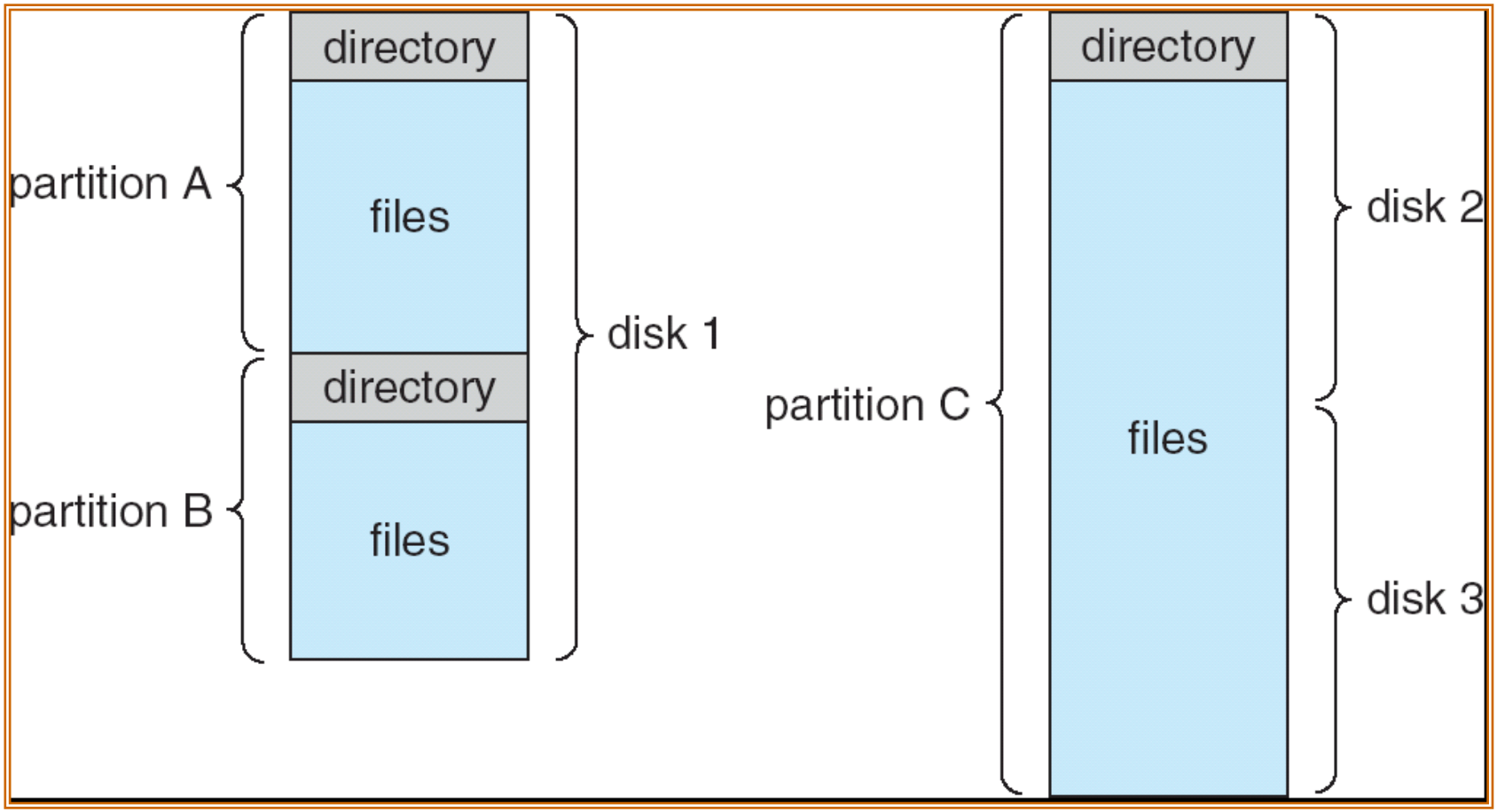


Tanto a estrutura de diretórios quando de arquivos residem no disco
Cópias de segurança dessas duas estruturas são mantidas em fitas

Estrutura de Disco

- Disco pode ser dividido em **partitições**
- Discos ou partições podem ser protegidas por **RAID** contra falhas
- Disco ou partição pode ser usada ***raw*** – sem um sistemas de arquivo, ou **formatada** com um sistema de arquivo
- Partições são conhecidas também como minidiscos ou *slices*
- Entidade que contém um sistema de arquivos é conhecido como **volume**
- Cada volume contendo um sistema de arquivos também mantém as informações deste em **diretório do dispositivo** ou **índice do volume**
- Assim como **sistemas de arquivos de propósito geral** existem muitos **sistemas de arquivos de propósito específico**, frequentemente todos dentro do mesmo sistema operacional ou computador

Uma Organização típica de Sistemas de Arquivos



Operações Realizadas em um Diretório

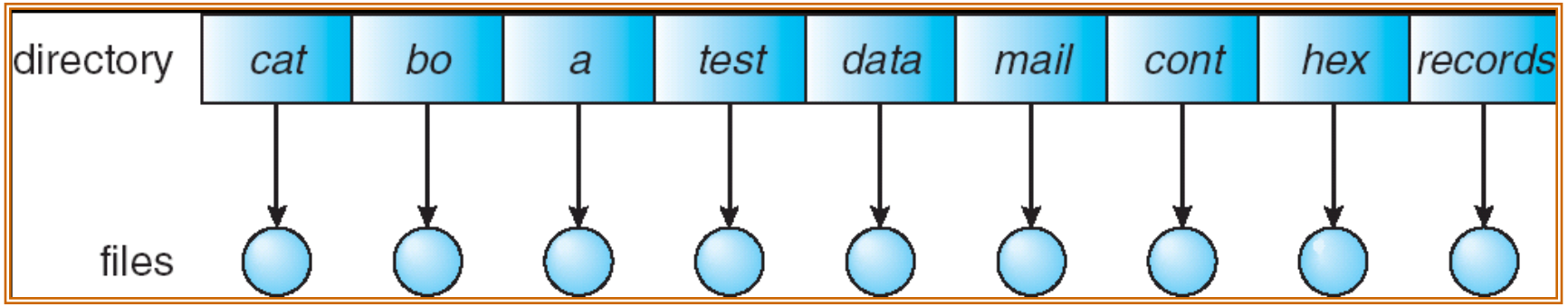
- Procurar por um arquivo
- Criar um Arquivo
- Excluir um Arquivo
- Listar um diretório
- Alterar o nome de um arquivo
- Percorrer o Sistema de Arquivos

Organizar o Diretório (Logicamente) para obter

- Eficiência – localizar um arquivo rapidamente.
- Nomeação – conveniência para usuários.
  - Dois usuários podem ter o mesmo nome para arquivos diferentes.
  - O mesmo arquivo pode ter vários nomes diferentes.
- Agrupamento – agrupamento lógico de arquivos por propriedades (ex.: todos programas em Java, todos jogos, …)

Diretório de um Nível

- Um único diretório para todos usuários

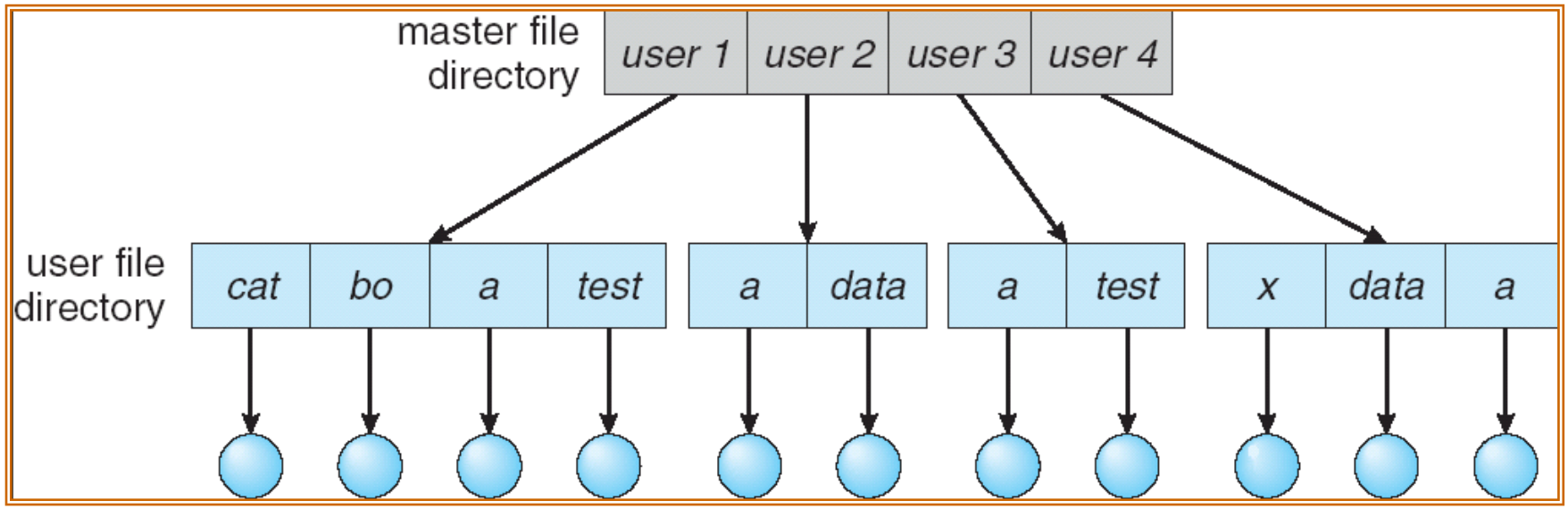


Problema de nomeação

Problema de agrupamento

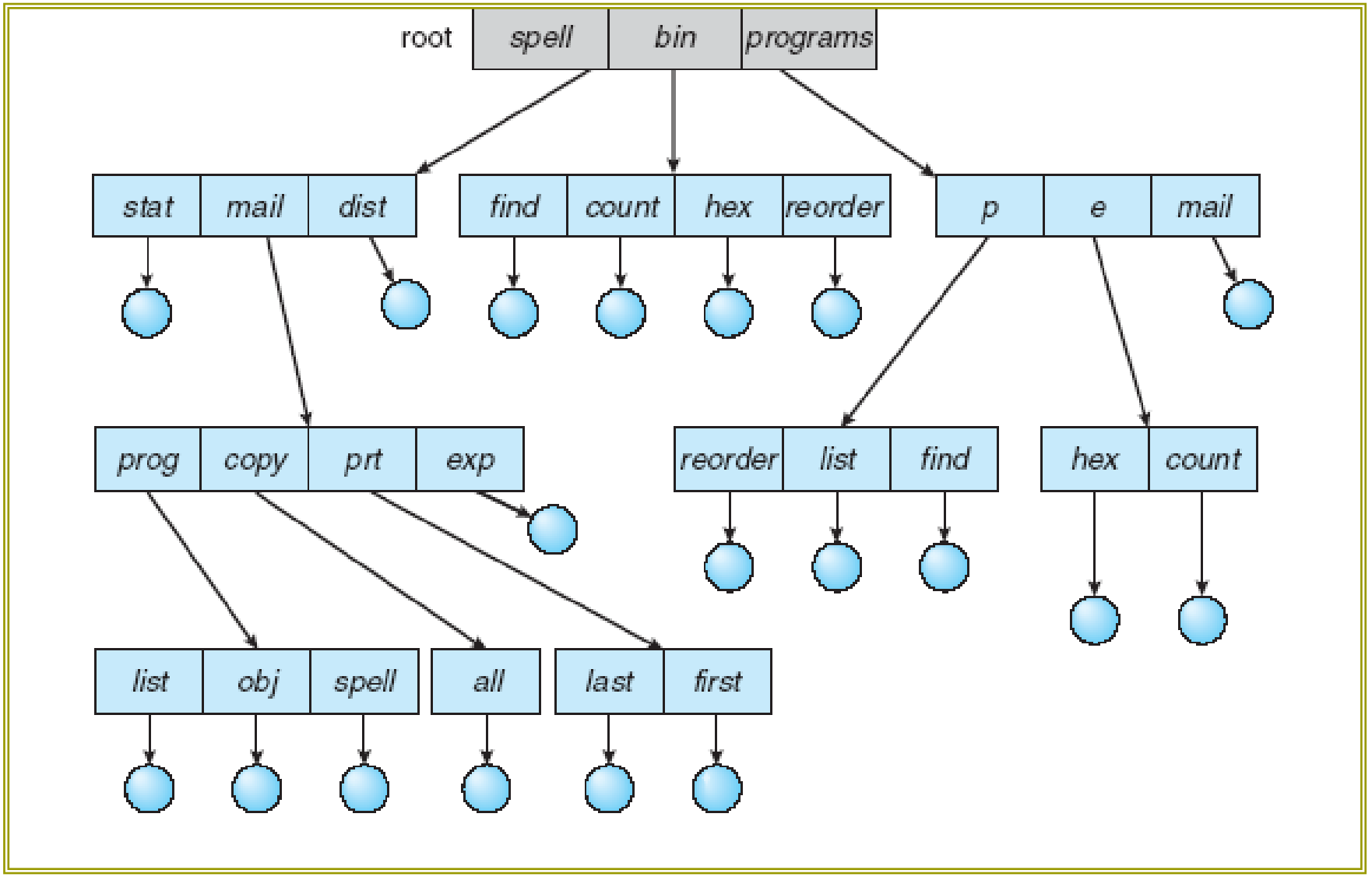
Diretório de dois Níveis

- Diretórios separados para cada usuário



- Caminho (*Path name*)
- Pode ter o mesmo nome de arquivo para diferentes usuários
- Procura eficiente
- Sem capacidade de agrupamento

Diretório com Estrutura de Árvore



Diretório com Estrutura de Árvore (Cont)

- Procura eficiente
- Capacidade de Agrupamento
- Diretório Corrente (diretório de trabalho)
  - cd /spell/mail/prog
  - type list

Diretório com Estrutura de Árvore (Cont)

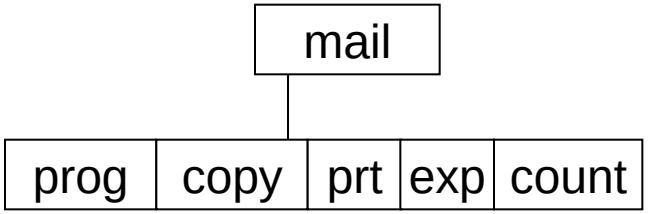
- Caminho **absoluto** ou **relativo**
- Criação de arquivos novos é feita no diretório corrente.
- Apagar um arquivo

  rm <file-name>
- Criação de novos subdiretórios é feita no diretório corrente.

  mkdir <dir-name>

  Exemplo: se o diretório corrente é /spell/mail

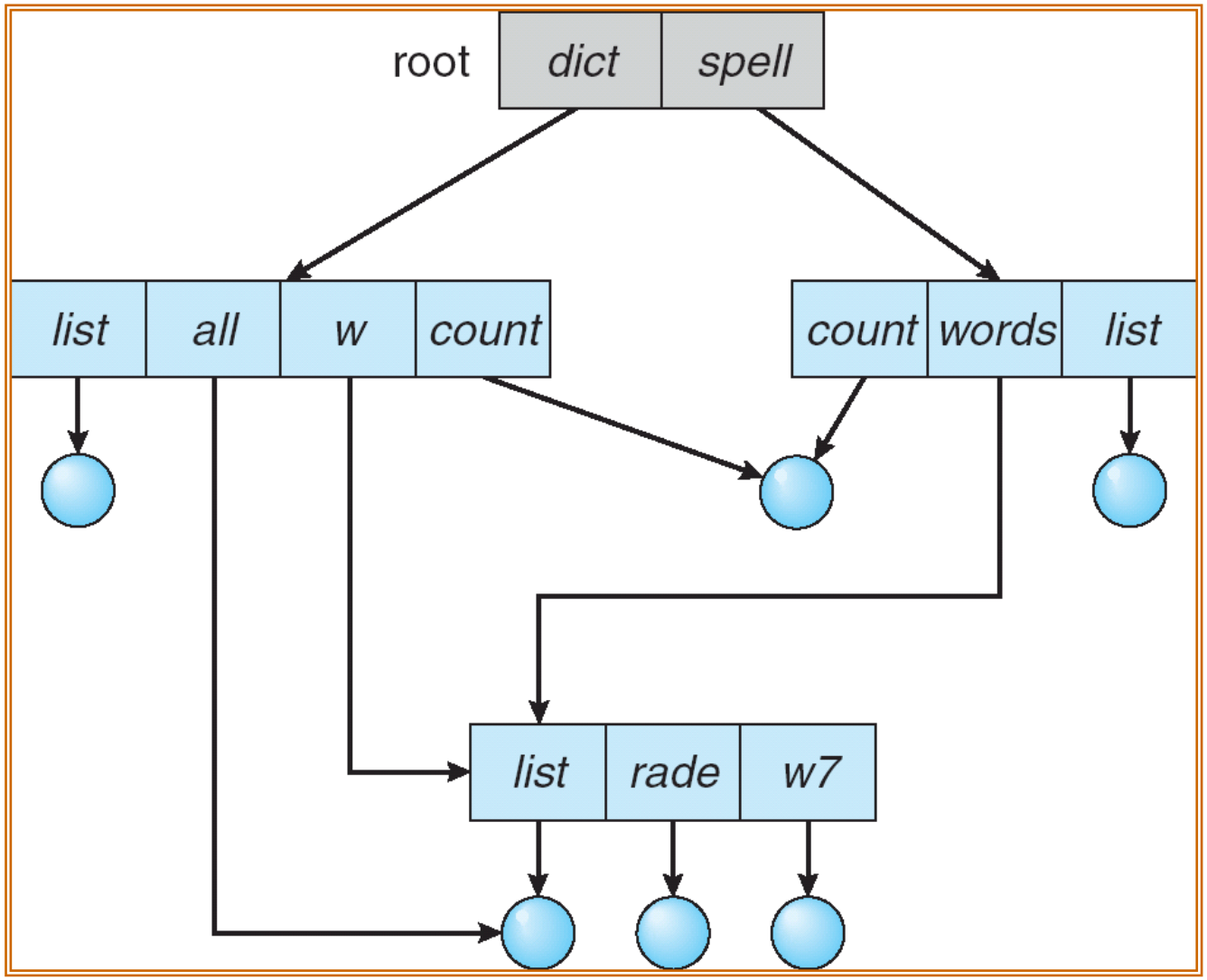
  mkdir count



Apagar "mail" ⇒ apaga toda a subárvore com a raiz "mail"

Diretórios com Estrutura de Grafo Acíclico

- Possui subdiretórios e arquivos compartilhados

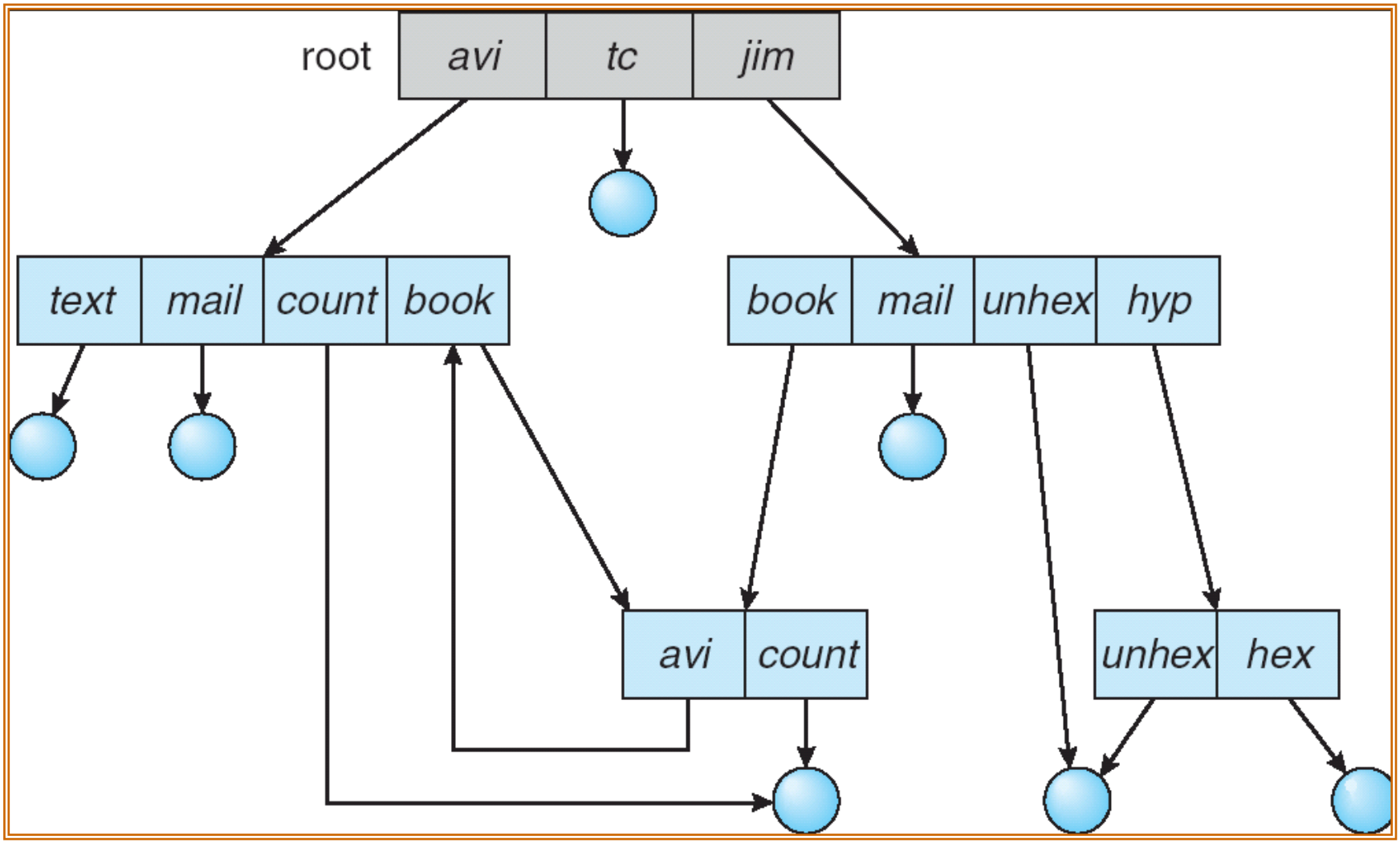


Diretórios com Grafo Acíclico (Cont.)

- Dois nomes diferentes (*aliasing*)
- Se *dict* apaga *list* $\Rightarrow$ ponteiro perigoso.

  Soluções:
  - Lista de referências a arquivos, então só podemos apagar todas as referências. Tamanho variável dos registros é um problema.
  - Lista de referências a arquivos, usando uma organização encadeada.
  - Contador do número de referências (*Entry-hold-count*).
- Tipo de entrada de novo diretório
  - **Link** – outro nome (ponteiro) para um arquivo existente
  - **Resolver o link** – seguir o ponteiro para localizar o arquivo

Diretórios com Estrutura de Grafo Geral



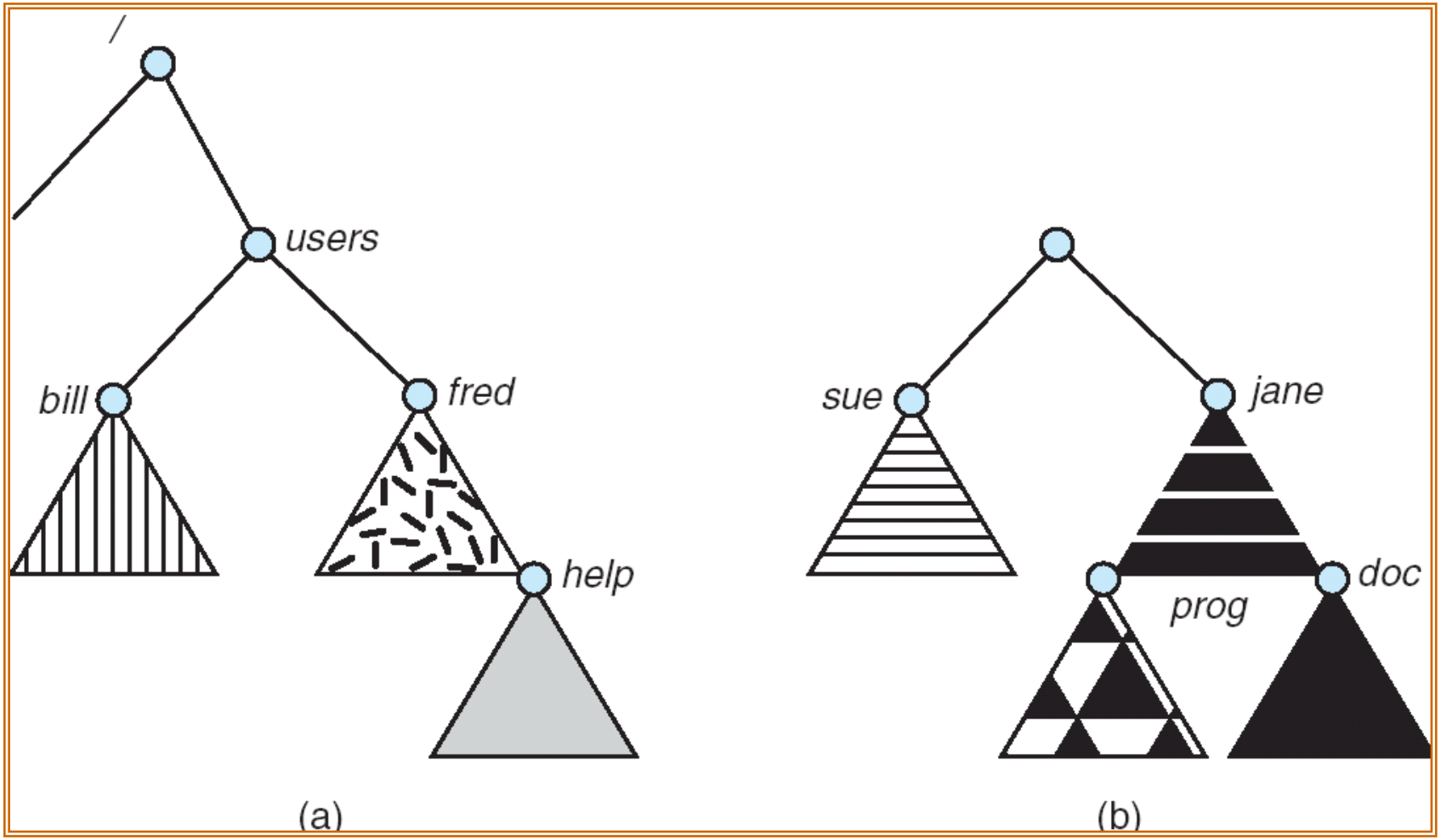
Diretórios com Grafo Geral (Cont.)

- Como garantir a não ocorrência de ciclos?
  - Permitir ligações para arquivos e não para subdiretórios.
  - Coletor de Lixo (*Garbage collection*).
  - Toda vez que uma nova ligação é adicionada usar um algoritmo de detecção de ciclo para determinar se é possível a ligação.

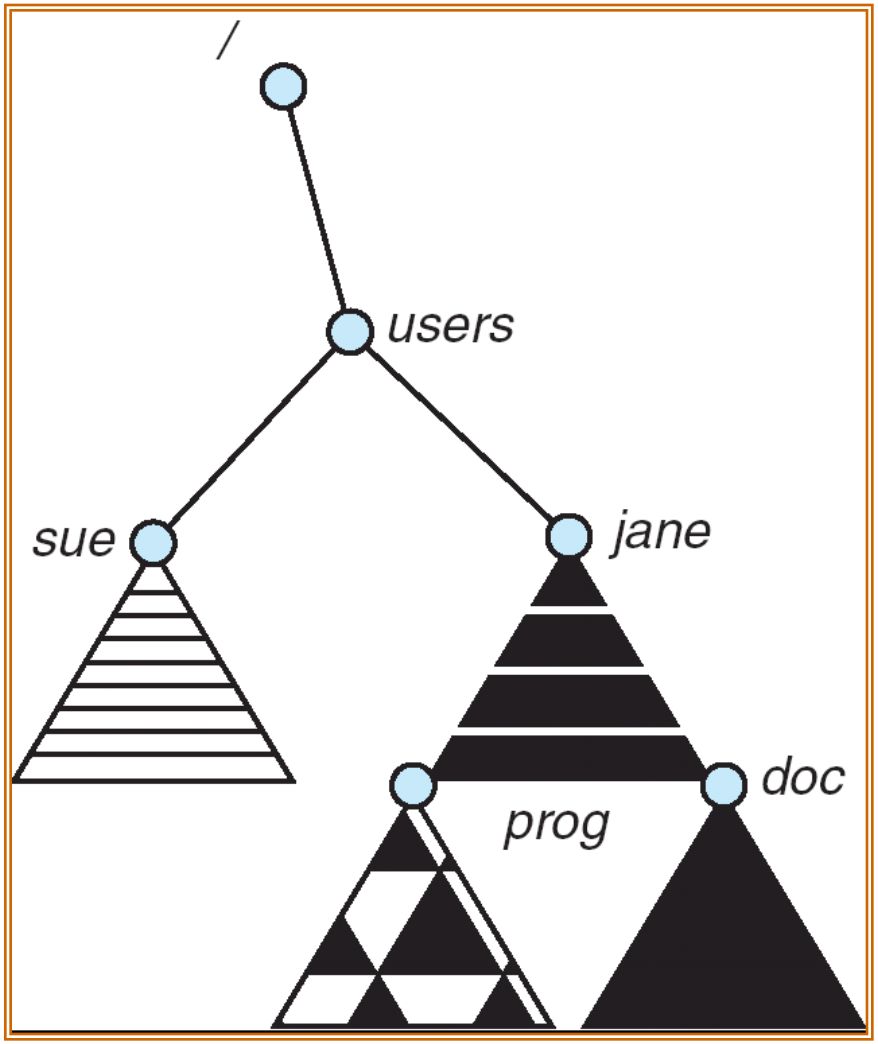
Montagem de Sistemas de Arquivos

- Um sistema de arquivos deve ser **montado** (*mount*) antes de ser acessado
- Um sistema de arquivos não montado (ex. Fig. 11-11(b)) é montado em um **ponto de montagem** (*mount point*)

(a) Existente (b) Partição não montada



Ponto de Montagem



Compartilhamento de Arquivos

- Compartilhamento de arquivos em sistemas multi-usuários é desejável
- Compartilhamento pode ser feita por um mecanismo de **proteção**
- Em sistemas distribuídos, arquivos podem ser compartilhados pela rede
- *Network File System* (NFS) é um método comum de compartilhamento de arquivos distribuídos

Compartilhamento de Arquivos – Vários Usuários

- ***User IDs*** identificam os usuários, possibilitando permissões e proteções por usuário
- **Group IDs** permitem usuários estarem em grupos, possibilitando diretos de acessos por grupo

Compartilhamento de Arquivos – Sistemas de Arquivos Remotos (*Remote File Systems*)

- Usa a rede para permitir acesso a arquivos entre sistemas
  - Manualmente via programas como FTP
  - Automaticamente, usando **sistemas de arquivos distribuídos**
  - Semi-automaticamente via web
- modelo **Cliente-servidor** permite clientes montar sistemas de arquivos remotos de servidores
  - Servidor pode atender múltiplos clientes
  - Identificação do Cliente e usuário no cliente é insegura ou complicada
  - **NFS** é o protocolo padrão cliente-servidor no UNIX para compartilhamento de arquivos
  - **CIFS** é o protocolo padrão no Windows
  - Chamadas ao sistemas de arquivo convencionais são traduzidas em chamadas remotas
- Sistemas distribuídos de Informações (***distributed naming services***) como LDAP, DNS, NIS, *Active Directory* implementam acesso unificado a informações necessárias para uso remoto

Compartilhamento de Arquivos
Modos de Falhas

- Sistemas de arquivos remotos adicionam novos modos de falhas, devido a falhas de rede e de servidor
- Recuperação de falhas pode envolver informações de estados sobre o status de cada requisição remota
- Protocolos sem estados (*stateless*) como o NFS incluem todas as informações em cada requisição, permitindo recuperação fácil ao custo de menor segurança

Compartilhamento de Arquivos
Semântica de Consistência

- **Semântica de consistência** especifica como múltiplos usuários estão acessando um arquivo compartilhado simultaneamente
  - Similar aos algortimos de sincronização de processos do Cap. 7
    - Tendem a ser menos complexos devido ao E/S de disco e a latência de rede (para sistemas de arquivos remotos)
  - *Andrew File System* (AFS) implementa semântica de compartilhamento complexa para arquivos remotos
  - *Unix file system* (UFS) implementa:
    - Escritas para um arquivo aberto são visíveis imediatamente para outros usuários do mesmo arquivo
    - Ponteiro para arquivo compartilhado permite múltiplos usuários ler e escrever concorrentemente
  - AFS tem semântica de sessão
    - Escritas só são visíveis em sessões que começam após o arquivo ter sido fechado

Proteção

- Dono/Criador do arquivo deve estar apto a controlar:
  - O que pode ser feito
  - Por quem

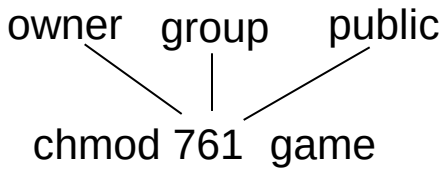
- Tipos de Acesso
  - **Leitura**
  - **Escrita**
  - **Execução**
  - **Adição** (*Append*)
  - **Exclusão**
  - **Listagem**

Listas de Acesso e Grupos

- Modos de acesso: leitura (***r**ead*), escrita(***w**rite*), execução (*e**x**ecute*)
- Três classes de usuários

			RWX
a) **acesso de dono** (*owner*)	7	⇒	1 1 1
b) **acesso de grupo** (*group*)	6	⇒	RWX 1 1 0
c) **acesso público** (*public*)	1	⇒	RWX 0 0 1

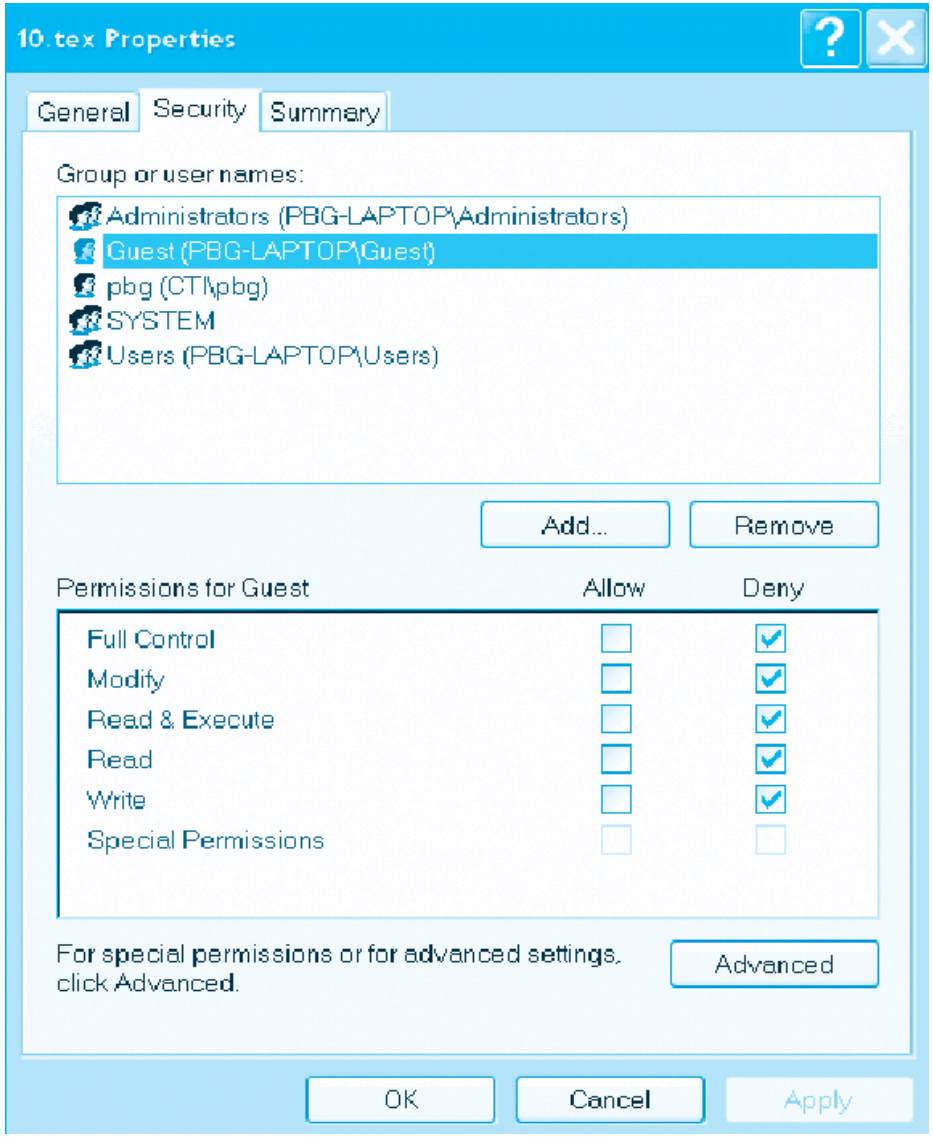
- Peça para o administrador criar um grupo (nome único), digamos G, e adicionar alguns usuários ao grupo.
- Para um arquivo ou subdiretório particular (digamos *game*), defina um acesso apropriado.



Associe um grupo a um arquivo

chgrp G game

Gerencimaneto da Lista de Controle de Acesso no Windows XP



Um exemplo de Listagem de Diretório no Unix

-rw-rw-r--	1 pbg	staff	31200	Sep 3 08:30	intro.ps
drwx------	5 pbg	staff	512	Jul 8 09.33	private/
drwxrwxr-x	2 pbg	staff	512	Jul 8 09:35	doc/
drwxrwx---	2 pbg	student	512	Aug 3 14:13	student-proj/
-rw-r--r--	1 pbg	staff	9423	Feb 24 2003	program.c
-rwxr-xr-x	1 pbg	staff	20471	Feb 24 2003	program
drwx--x--x	4 pbg	faculty	512	Jul 31 10:31	lib/
drwx------	3 pbg	staff	1024	Aug 29 06:52	mail/
drwxrwxrwx	3 pbg	staff	512	Jul 8 09:35	test/

Fim do Capítulo 10