SISTEMAS OPERACIONAIS

AULA 16 – GERENCIAMENTO DE ARQUIVOS, PARTE 1

Prof.ª Sandra Cossul, Ma.

INTRODUÇÃO

- **Arquivos surgem com a necessidade de armazenar informações para uso posterior!**

- Parte importante do uso do computador: recuperar e apresentar informações previamente armazenadas (documentos, fotos, músicas e vídeos)

- SO também mantém informações armazenadas para uso posterior: programas, bibliotecas e configurações

INTRODUÇÃO

- O **sistema de arquivos** é a parte mais visível de um SO
- Provê o mecanismo para **armazenamento** e **acesso** de **dados** e **programas**

- O sistema de arquivos consiste de duas partes:
  - **Coleção de arquivos –** em que cada um armazena dados
  - **Estrutura de diretórios –** que organiza e provê informações sobre todos os arquivos no sistema

SISTEMAS GERENCIADORES DE ARQUIVOS

- Programas **utilitários** que rodam como **parte do SO**

- Estes tem como objetivo permitir aos usuários **acessar e salvar arquivos**, mantendo a **integridade do conteúdo** destes.

- Sistemas de arquivos admitem ao usuário criar arquivos permitindo:
  - **Armazenamento –** arquivos são guardados de forma permanente na memória (HD, SSD)
  - **Compartilhamento –** cada arquivo tem um nome e pode ter diferentes permissões de acesso
  - **Estrutura –** arquivos podem ter uma estrutura interna de acordo com a aplicação

SISTEMAS GERENCIADORES DE ARQUIVOS

- Um **sistema de arquivos** pode ser visto como uma imensa **estrutura de dados** que indica como os arquivos devem ser gravados e lidos pelo SO do computador.

- **Responsável pela manipulação de dados de um dispositivo de armazenamento!**

- **Exemplos**:
  - NTFS – sistemas Windows
  - ext2/ext3/ext4 – Linux
  - HPFS – MacOS
  - FAT32 – uso em memórias flash (cartões de memória, pendrives)

NTFS X FAT32

- **NTFS**
  - Amplamente utilizado nos SOs da Microsoft
  - Suporta recuperação de dados em caso de falhas
  - Suporta redundância de dados (replicação)
  - Suporte a criptografia
  - Possui um esquema de permissões de acesso (segurança)

  - Desvantagem: falta de compatibilidade com outros SOs
  - Desvantagem: mais lento que o FAT (devido as diretivas de segurança e acesso)
  - Indicado para dispositivos de armazenamento não removíveis (HD e SSD)

NTFS X FAT32

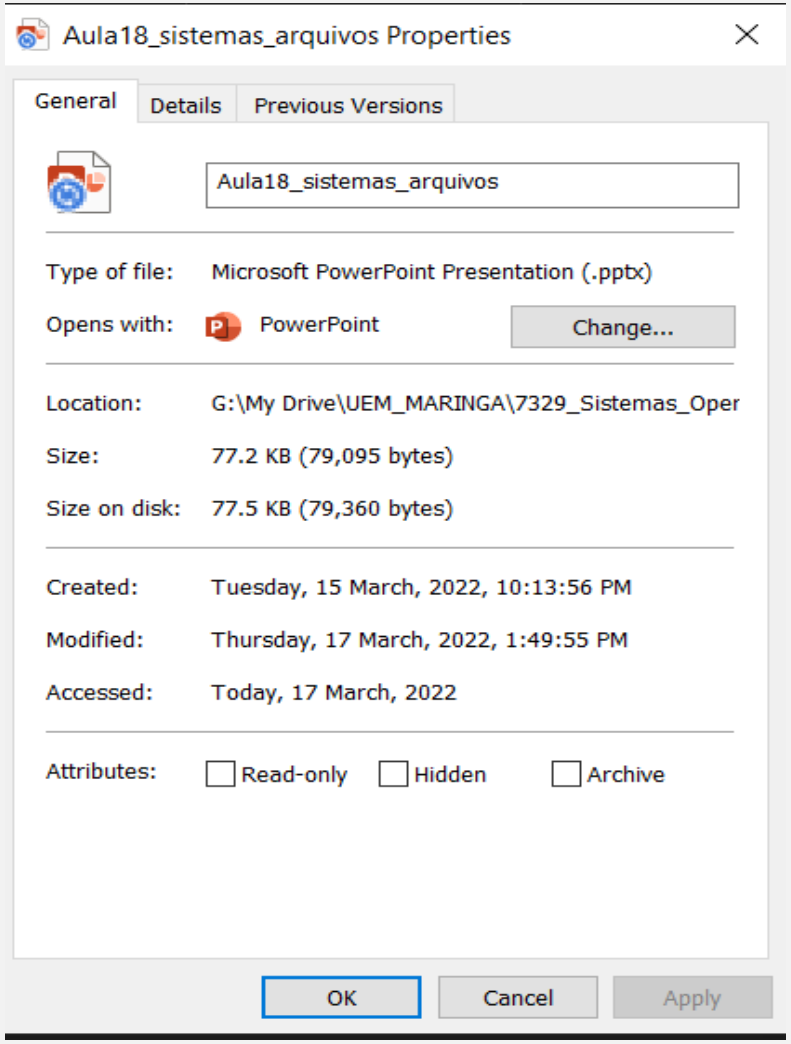
- **FAT32 – File Allocation Table**
  - Sistema de arquivos mais antigo
  - Principal vantagem: compatibilidade entre diversos SOs e dispositivos USB
  - Limitações: desperdício de memória (utiliza unidades de alocação – clusters com tamanhos fixos)
  - Limitações: suporta arquivos de até 4GB
  - Indicado para dispositivos de armazenamento móveis (como pen drives) para compartilhamento de arquivos menores

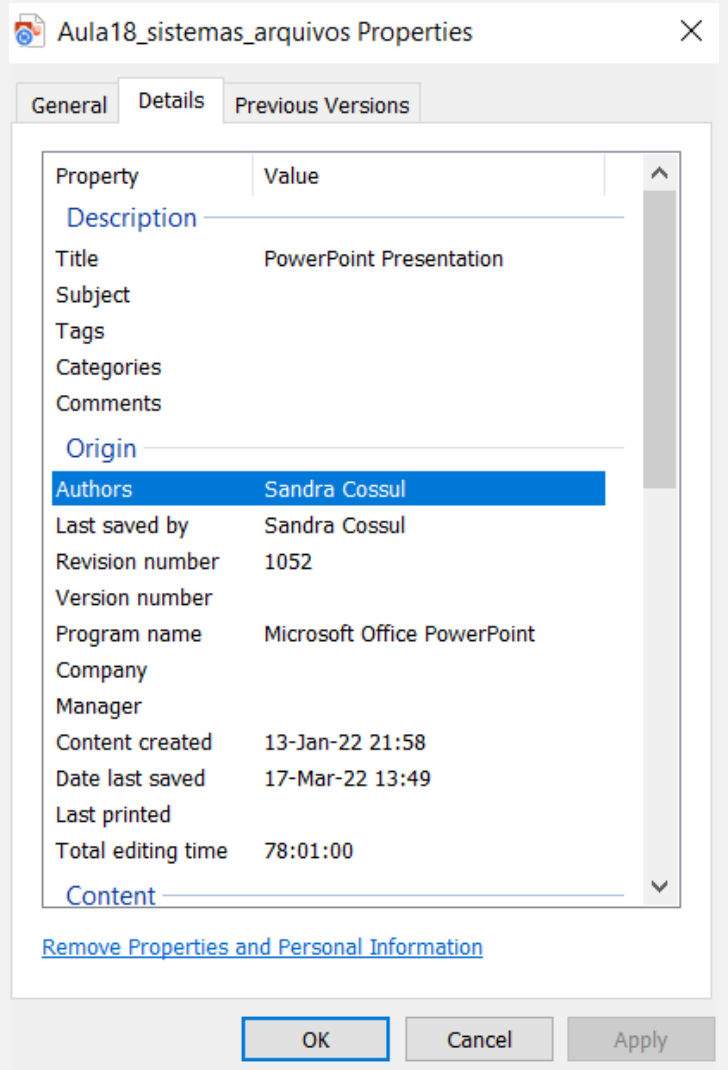
ATRIBUTOS DE ARQUIVOS

- **Nome –** nome simbólico do arquivo (visível ao usuário)
- **Identificador –** tag identificadora, normalmente um número (não-visível ao usuário)
- **Tipo –** tipo do arquivo (.docx, .pdf, .jpeg)
- **Localização –** ponteiro para a localização do arquivo no dispositivo
- **Tamanho –** tamanho do arquivo (em bytes, palavras ou blocos) e tamanho máximo permitido
- **Proprietário –** criador do arquivo
- **Proteção –** informações de controle de acesso determina quem pode ler, escrever, executar, etc.
- **Timestamp e identificação de usuário –** informações mantidas sobre criação, última modificação e último uso do arquivo. Dados úteis para proteção, segurança e monitoramento de uso.

ATRIBUTOS DE ARQUIVOS

Exemplo de atributos de arquivos no Windows:





OPERAÇÕES EM ARQUIVOS

- As aplicações e o SO usam arquivos para armazenar e recuperar dados

- O **acesso aos arquivos** é feito através de um **conjunto de operações**, geralmente implementadas sob a forma de **chamadas de sistema e funções de bibliotecas**

OPERAÇÕES EM ARQUIVOS

- **Criar**
  - Alocar espaço na memória para o arquivo
  - Criar uma entrada para o novo arquivo no diretório
  - Definir valores para seus atributos (nome, localização, data, permissões, etc.)

- **Abrir**
  - Permitir que um processo execute funções no arquivo
  - Todas operações, exceto criar e deletar, requerem primeiro abrir o arquivo

- **Escrever**
  - Transferir dados na memória da aplicação para o arquivo no dispositivo físico
  - Novos dados podem ser adicionados no final do arquivo ou sobrescrever dados já existentes

OPERAÇÕES EM ARQUIVOS

- **Ler**
  - Permite transferir dados presentes no arquivo para uma área de memória da aplicação
- **Deletar**
  - Procurar o diretório pelo nome do arquivo
  - Liberar a memória utilizada pelo arquivo e apagar o arquivo
- **Fechar**
  - Ao concluir o uso do arquivo, a aplicação devem informar ao SO que o mesmo não é mais necessário, liberando as suas estruturas de gerência
- **Alterar atributos**
  - Modificar valores dos atributos como nome, proprietário, permissões, etc.

OPERAÇÕES EM ARQUIVOS

- **Operações de Entrada e Saída**
  - Realizadas através de chamadas de sistema
  - Chamadas de sistema fornecem uma interface simples e uniforme entre a aplicação e os diversos dispositivos, permitindo leitura/gravação, criação/eliminação de arquivos.

TIPOS DE ARQUIVOS

- Um SO reconhece e suporta diversos tipos de arquivos

- A **extensão** é incluída junto com o **nome do arquivo**:
  - resumo.docx
  - exemplo.c
  - trabalho.pdf

- Os programas de aplicação geram os arquivos em determinado formato:
  - Word → .doc ou .docx
  - PowerPoint → .pptx

TIPOS/FORMATOS DE ARQUIVOS

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, perl, asm	source code in various languages
batch	bat, sh	commands to the command interpreter
markup	xml, html, tex	textual data, documents
word processor	xml, rtf, docx	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	gif, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	rar, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, mp3, mp4, avi	binary file containing audio or A/V information

INTERFACE DE ACESSO

- Interface de acesso a um arquivo é uma representação lógica do arquivo, denominada **descritor de arquivo,** e um conjunto de funções para manipular o arquivo.

- **Interface de baixo nível** → chamadas de sistema do SO
- **Interface de alto nível** → funções na linguagem de programação (biblioteca de funções)

INTERFACE DE ACESSO

- **Chamadas de sistema para arquivos**

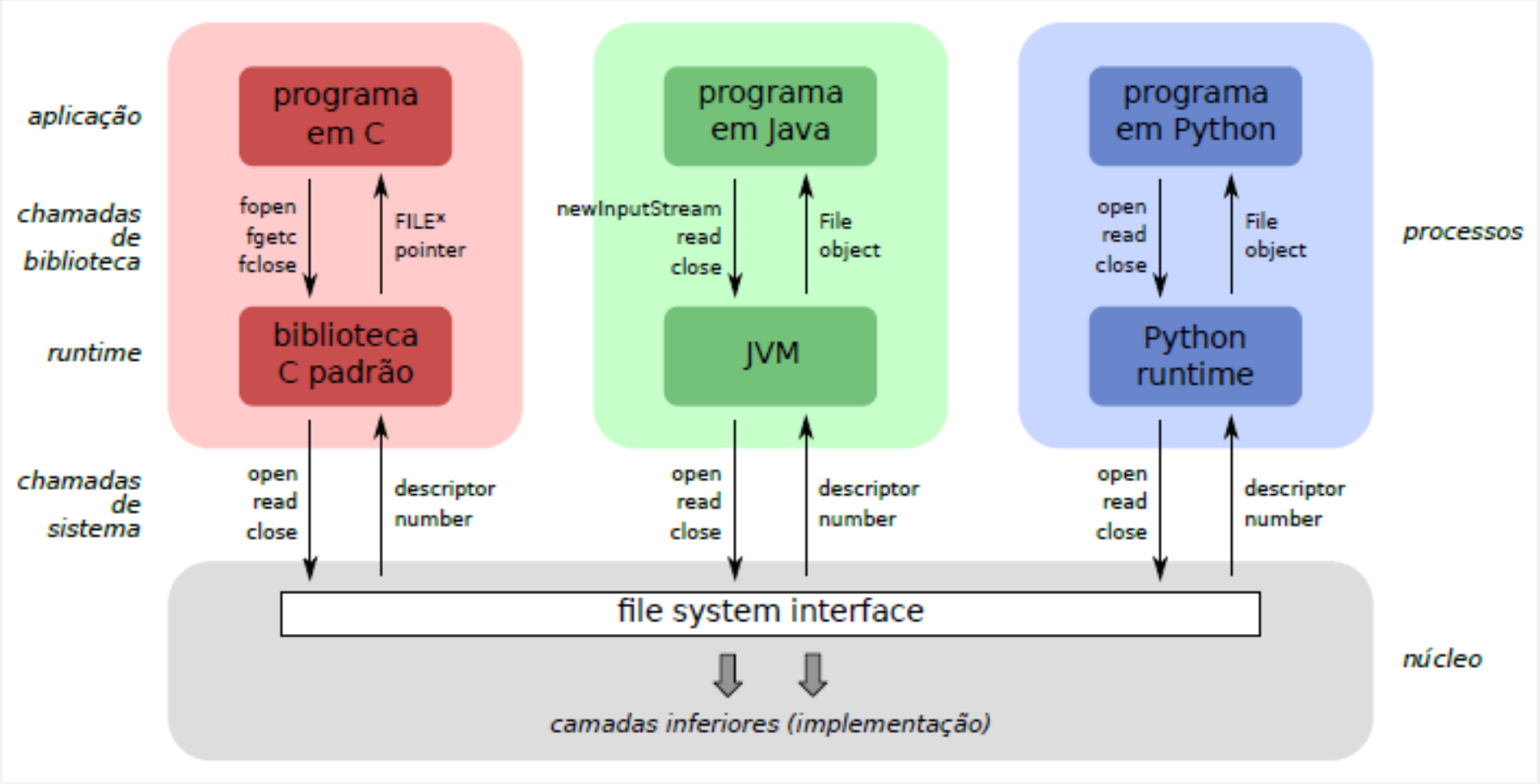
Operação	Linux	Windows
Abrir arquivo	OPEN	NtOpenFile
Ler dados	READ	NtReadRequestData
Escrever dados	WRITE	NtWriteRequestData
Fechar arquivo	CLOSE	NtClose
Remover arquivo	UNLINK	NtDeleteFile
Criar diretório	MKDIR	NtCreateDirectoryObject

- **Funções de biblioteca para arquivos**

Operação	C (padrão C99)	Java (classe **File**)
Abrir arquivo	fd = fopen(...)	obj = File(...)
Ler dados	fread(fd, ...)	obj.read()
Escrever dados	fwrite(fd, ...)	obj.write()
Fechar arquivo	fclose(fd)	obj.close()
Remover arquivo	remove(...)	obj.delete()
Criar diretório	mkdir(...)	obj.mkdir()

INTERFACE DE ACESSO

- **Código da aplicação:** chamadas de funções
- **Suporte de execução:** traduz as chamadas de função nas chamadas de sistema aceitas pelo SO
- **Chamadas de sistema:** sistema de arquivos do núcleo do SO, que encaminha para as rotinas com as ações



DESCRITORES DE ARQUIVOS– EXEMPLO ABERTURA DE UM ARQUIVO

- **No processo:**
  - a aplicação solicita a abertura do arquivo (fopen(), se for um programa C);
  - o **suporte de execução** da aplicação recebe a **chamada de função**, trata os parâmetros recebidos e **invoca uma chamada de sistema** para abrir o arquivo.

- **No núcleo:**
  - o **núcleo** recebe a **chamada de sistema**;
  - **localiza o arquivo no dispositivo físico**, usando seu nome e caminho de acesso;
  - **verifica** se o processo tem as **permissões necessárias** para usar aquele arquivo da forma desejada

DESCRITORES DE ARQUIVOS– EXEMPLO ABERTURA DE UM ARQUIVO

- **No núcleo** (continuação)**:**
  - **cria uma estrutura de dados** na memória do núcleo para representar o arquivo aberto;
  - **insere** uma **referência** a essa **estrutura** na relação de **arquivos abertos** mantida pelo núcleo, para fins de gerência;
  - **devolve** à aplicação uma **referência** a essa estrutura (o descritor de baixo nível), para ser usada nos acessos subsequentes ao arquivo.

- **No processo:**
  - o **suporte** de execução **recebe** do núcleo o **descritor** de **baixo nível** do arquivo;
  - o **suporte** de execução cria um **descritor** de **alto nível** e o devolve ao código da aplicação;
  - o **código da aplicação** recebe o descritor de alto nível do arquivo aberto, para usar em suas operações subsequentes envolvendo aquele arquivo.

ESTRUTURA DE ARQUIVOS

- **Tipos de arquivos** podem ser utilizados para indicar a **estrutura interna do arquivo.**
- Certos arquivos devem estar de acordo com certa estrutura para o SO conseguir interpretá-los.
- Cada **programa de aplicação** deve incluir seu próprio código para **interpretar um arquivo na sua estrutura** apropriada.
- No entanto, os SOs devem "entender" pelo menos uma estrutura, por exemplo, um arquivo executável, de forma que o sistema consiga carregar e executar programas.

ESTRUTURA DE ARQUIVOS

- Uma **aplicação pode definir um formato próprio** para armazenar seus dados **ou pode seguir formatos padronizados**

- A adoção de um **formato proprietário** ou exclusivo **limita o uso** das informações armazenadas, pois somente aplicações que reconheçam aquele formato específico conseguem ler corretamente as informações contidas no arquivo.

MÉTODOS DE ACESSO

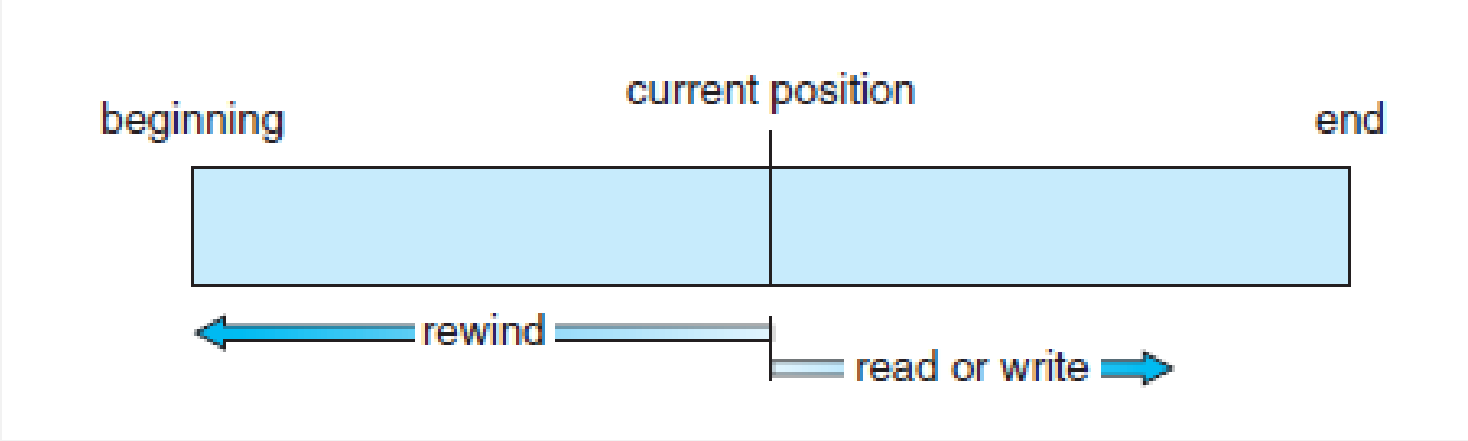
- Arquivos guardam informações.

- Quando utilizados, a informação deve ser acessada e trazida para a memória do computador

- A informação em um arquivo pode ser acessada de diferentes formas:
  - **Acesso sequencial**
  - **Acesso direto**
  - **Acesso indexado ou acesso por chave**
  - **Acesso mapeado em memória**

MÉTODOS DE ACESSO

- **Acesso sequencial**
  - forma mais simples e usual de acesso a arquivos, usada pela maioria das aplicações
  - Informação do arquivo é processada em ordem, um registro após o outro
  - Ponteiro de acesso – aponta para a primeira posição do arquivo
  - Chegada ao final do arquivo – flag EOF (*End-of-File*



MÉTODOS DE ACESSO

- **Acesso direto (ou relativo)**
  - Um arquivo é formado por registros de tamanho fixo
  - Programas fazem a leitura e escrita em qualquer ordem
  - Permite acesso aleatório (não tem restrição de ordem de leitura/escrita)
  - São úteis para acesso imediato de grandes quantidades de informações (como por exemplo, banco de dados) que precisam acessar rapidamente as posições do arquivo correspondentes aos registros
  - É necessário especificar o número do registro

MÉTODOS DE ACESSO

- **Acesso indexado ou Acesso por chave**
  - Método mais sofisticado e tem como base o acesso direto
  - O arquivo deve possuir uma área de índice onde existam ponteiros para os diversos registros
  - Quando a aplicação deseja acessar um registro, deverá ser especificada uma chave através da qual o sistema pesquisará, na área de índice, o ponteiro correspondente, a partir disso, acessa diretamente o arquivo.

MÉTODOS DE ACESSO

- **Acesso mapeado em memória**
  - Faz uso dos mecanismos de paginação em disco e memória virtual
  - O arquivo é associado a um vetor de registros de mesmo tamanho na memória principal
    - Cada posição do vetor corresponde à sua posição equivalente no arquivo (referência para o arquivo)
  - Usado pelo SO para carregar código executável (programas e bibliotecas) na memória, sob demanda.
  - Os dados são lidos do arquivo para a memória em páginas.

COMPARTILHAMENTO DE ARQUIVOS

- Sistema multitarefas → arquivos acessados por mais de um processo, ou mesmo mais de um usuário

- Acesso simultâneo pode gerar condições de disputa (corrida) que podem levar à inconsistência de dados

- **Acesso concorrente em <u>leitura</u> a um arquivo** → não gera problema
- **Acesso concorrente em <u>leitura e escrita</u>** → pode levar a condições de disputa

COMPARTILHAMENTO DE ARQUIVOS

- **Solução –** Travas
- uso de travas de exclusão mútua (*mutex locks*), como já estudamos

- A maioria dos SOs oferece algum **mecanismo de sincronização para o acesso a arquivos**, na forma de uma ou mais **travas** (*locks*) associadas a cada **arquivo aberto**

- A **sincronização** pode ser feita sobre o **arquivo inteiro** ou sobre algum **trecho específico**

- As **travas de arquivos** são atribuídas a **processos.** Serão liberadas quando o processo fecha o arquivo ou encerra sua execução.

CONTROLE DE ACESSO

- É importante definir claramente o **proprietário de cada arquivo** e que **operações** ele e outros usuários do sistema podem efetuar sobre o mesmo;

- ***Proprietário* -** identifica o usuário dono do arquivo, geralmente aquele que o criou
- ***Permissões de acesso* -** define que operações cada usuário do sistema pode efetuar sobre o arquivo

PRÓXIMA AULA

- **Sistemas de arquivos**
- **Diretórios de arquivos**
- **Proteção**

BIBLIOGRAFIA

- Tanenbaum, A. S. **Sistemas Operacionais Modernos.** Pearson Prentice Hall. 3rd Ed., 2009.
- Silberschatz, A; Galvin, P. B.; Gagne G.; **Fundamentos de Sistemas Operacionais.** LTC. 9th Ed., 2015.
- Stallings, W.; **Operating Systems: Internals and Design Principles**. Prentice Hall. 5th Ed., 2005.
- Oliveira, Rômulo, S. et al. **Sistemas Operacionais** - V11 - UFRGS. Disponível em: Minha Biblioteca, Grupo A, 2010.