Curso de Engenharia de Computação
Sistemas Operacionais

Entrada e Saída – Parte II

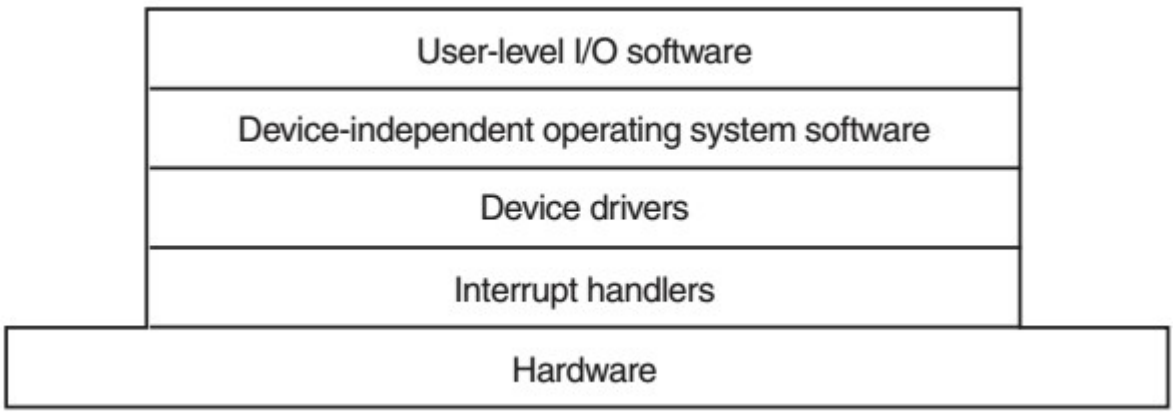
Slides da disciplina Sistemas Operacionais
Curso de Engenharia de Computação
Instituto Mauá de Tecnologia – Escola de Engenharia Mauá
Prof. Marco Antonio Furlan de Souza

Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA

MAUÁ

- **Camadas**



Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA
MAUÁ

- **Manipuladores de interrupção**
  - Quando uma **interrupção** de E/S **ocorre**, ela provoca a **ativação** de um **driver**, que **normalmente** é **estruturado** como um **processo** que **executa** no **kernel** (com estados, pilha e contadores de programa próprios);
  - Existem diversos passos que devem ser executados quando ocorre uma interrupção:

1. Salvar qualquer registrador (incluindo PSW) que não tenha sido salvo pelo hardware de interrupção.
2. Definir um contexto (dados) para o procedimento de serviço de interrupção (TLB, MMU e tabela de páginas).
3. Definir uma pilha para o procedimento de serviço de interrupção.
4. Alertar o controlador de interrupção.
5. Copiar os registradores de onde eles foram salvos (alguma pilha) para a tabela de processo.
6. Executar o procedimento de serviço de interrupção, que extrairá informação dos registradores do dispositivo que provocou a interrupção.
7. Escolher que processo executar na sequência.
8. Ajustar a MMU para o processo que será executado (pode envover TLB)
9. Carregar os registradores do novo processo, incluindo PSW.
10. Iniciar a execução do novo processo.

Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA

- **Device drivers**
  - Cada **controlador de dispositivo** possui **um** ou **mais registradores de dispositivo** que são utilizados para **comandá-lo**, para **ler** seu **estado** ou ambos;
  - O **número** de **registradores** de dispositivo e a **natureza** dos comandos **varia radicalmente** de dispositivo para dispositivo. **Exemplos**:
    - **Mouse**: precisa armazenar a sua posição e o estado dos botões;
    - **Disco rígido**: precisa armazenar informações sobre setores, trilhas, cilindros, cabeças, movimento do braço, motor, tempos de assentamento da cabeça etc.

Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA
MAUÁ

- **Device drivers**
  - O **código** do sistema operacional que **controla** um **dispositivo** (pelo seu controlador) é denominado de **device driver** e normalmente é fornecido pelo fabricante do dispositivo;
  - **Cada device driver controla** um **tipo** de **dispositivo** ou uma **classe** de **dispositivos** relacionados;
  - Existem **dispositivos muito diferentes** que são **baseados** na **mesma tecnologia** subjacente. Por **exemplo**, **USB** (*Universal Serial Bus*) – utiliza uma organização em camadas similar às camadas de rede – na camada mais alta ainda há a separação do tipo de dispositivo.

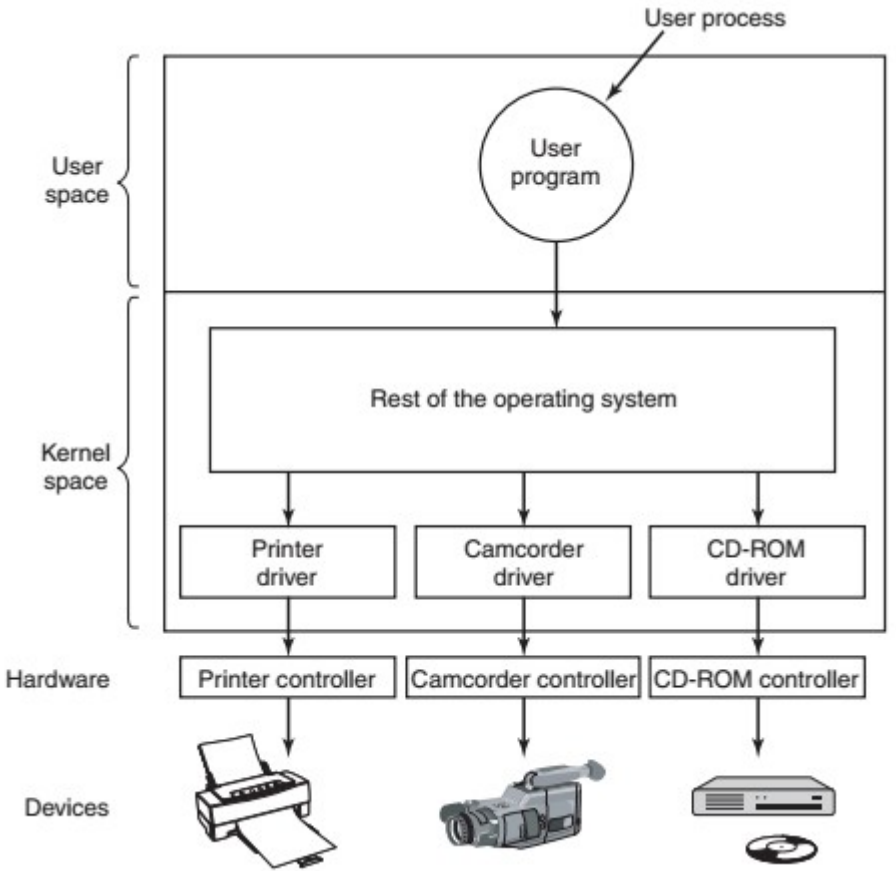
Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA
MAUÁ

- **Device drivers**
  - Para **acessar** o **hardware** de um **dispositivo** (seus registradores) **normalmente** o **device driver** deve ser **parte** do **kernel** do sistema operacional;
  - Mas é **possível construir device drivers** que **executam** no **espaço do usuário**, com chamadas para leitura e escrita nos registradores dos dispositivos. Esta **abordagem isola** o **kernel** dos **drivers** e os **drivers** de outros drivers, **eliminado crashes** que acabam interferindo no kernel - o MINIX3 adota esta abordagem;
  - Os **device drivers** podem ser **classificados** ainda em **devices** de **bloco** e **devices** de **caractere** e podem estar **compilados** no **kernel** ou serem **carregados dinamicamente**.

Camadas de software de E/S

- **Device drivers**



Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA

- **Device drivers**
  - **Funcionamento típico**
    - O **device driver inicia verificando** os **parâmetros** de **entrada** para certificar-se que são **válidos**. Se não, um erro é retornado. Se sim, o driver traduz as características do dispositivo real em termos desses parâmetros;
    - O **driver verifica se** o **dispositivo** está em **uso**. Se sim, a requisição será enfileirada para processamento posterior. Senão, o estado do hardware é verificado para saber se a requisição pode ser feita ou não. Quando o **dispositivo** está **pronto**, o **controle** se **inicia**;
    - Controlar **significa gerar** uma **série** de **comandos** para o **dispositivo**. A **cada comando** enviado é necessário **certificar** que o **dispositivo** o **recebeu** e está **pronto** para **receber outro**.

Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA

- **Device drivers**
  - **Funcionamento típico**
    - **Após** o **envio** dos **comandos**, pode **ocorrer** um dos dois casos: **(a)** O **device driver** deve **aguardar** até o **controlador executar** o **trabalho** para ele, sendo avisado por uma interrupção para desbloqueá-lo; **(b)** A **operação finaliza sem atraso** e **não** é necessário que o **device driver** seja **bloqueado**;
    - Por fim, ele **retorna** algum **valor** de **informação** de **estado** para quem o solicitou;
    - Na realidade, o **funcionamento** é mais **complexo** – **reentrância** (uma solicitação ao driver enquanto ele processa outra), utilização de dispositivos **hot-plug**, etc;
    - **Nota**: device drivers não executam chamadas de sistema!

Camadas de software de E/S

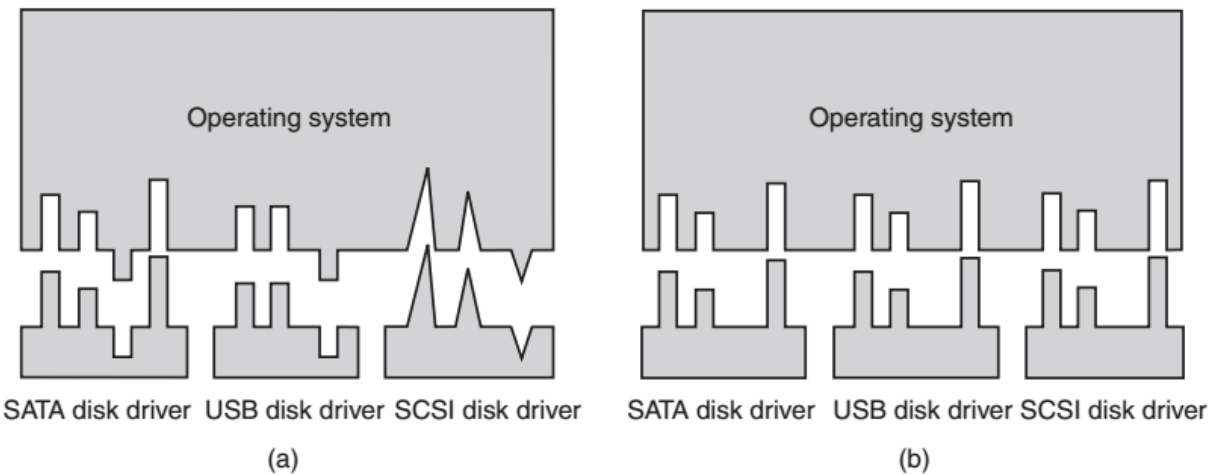
INSTITUTO MAUÁ DE TECNOLOGIA
MAUÁ

- **Software de E/S independente de dispositivo**
  - A **função básica** de **software** de **E/S independente** de **dispositivo** é **executar** as **funções** de **E/S** que são **comuns** a **todos** os **dispositivos** e **prover** uma **interface uniforme** aos **programas** do **nível** de **usuário**;
  - **Funcionalidades**:

Uniform interfacing for device drivers
Buffering
Error reporting
Allocating and releasing dedicated devices
Providing a device-independent block size

Camadas de software de E/S

- **Software de E/S independente de dispositivo**
  - **Interface uniforme para device drivers**
    - **Questão importante**: como fazer com que todos dispositivos de E/S e drivers sejam parecidos ao siistema operacional? Ter que inventar novos meios de utilização para novos dispositivos não é uma boa ideia…



(a) Interfaces distintas
(b) Mesma interface

Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA

- **Software de E/S independente de dispositivo**
  - **Interface uniforme para device drivers**
    - Para se **conseguir** a **uniformidade**, para cada **classe** de **dispositivos** (discos, impressoras etc) o **sistema operacional** deve **definir** um **conjunto** de **funções** que o **device driver** deve **fornecer**;
    - Por **exemplo**, para um **disco** isso inclui leitura e escrita, ligar e desligar, formatação etc.
    - Em geral, um **device driver mantém** uma **tabela** de **ponteiros** para essas **funções**. Quando o **driver** é **carregado**, o **sistema operacional armazena** o **endereço** desta **tabela** de **ponteiros** de funções de **modo** que, **quando ele precisa chamar** uma dessas **funções** ele pode **realizar** uma **chamada** indireta **por** esta **tabela**. Esta é a **interface** entre o **driver** o o **resto** do **sistema operacional**.

Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA

- **Software de E/S independente de dispositivo**
  - **Interface uniforme para device drivers**
    - Outro **aspecto** de **uniformização** é a **nomenclatura** dos **dispositivos** de E/S;
    - Por **exemplo**, no **UNIX** um **nome de dispositivo** como `/dev/disk0` **identifica unicamente** o **i-node** para um **arquivo especial**, que **contém** o **número maior do dispositivo**, que é utilizado para **localizar** o **driver**. Também **contém** o **número menor do dispositivo**, que é **passado** como **parâmetro** ao driver para identificar a unidade a ser lida ou escrita.

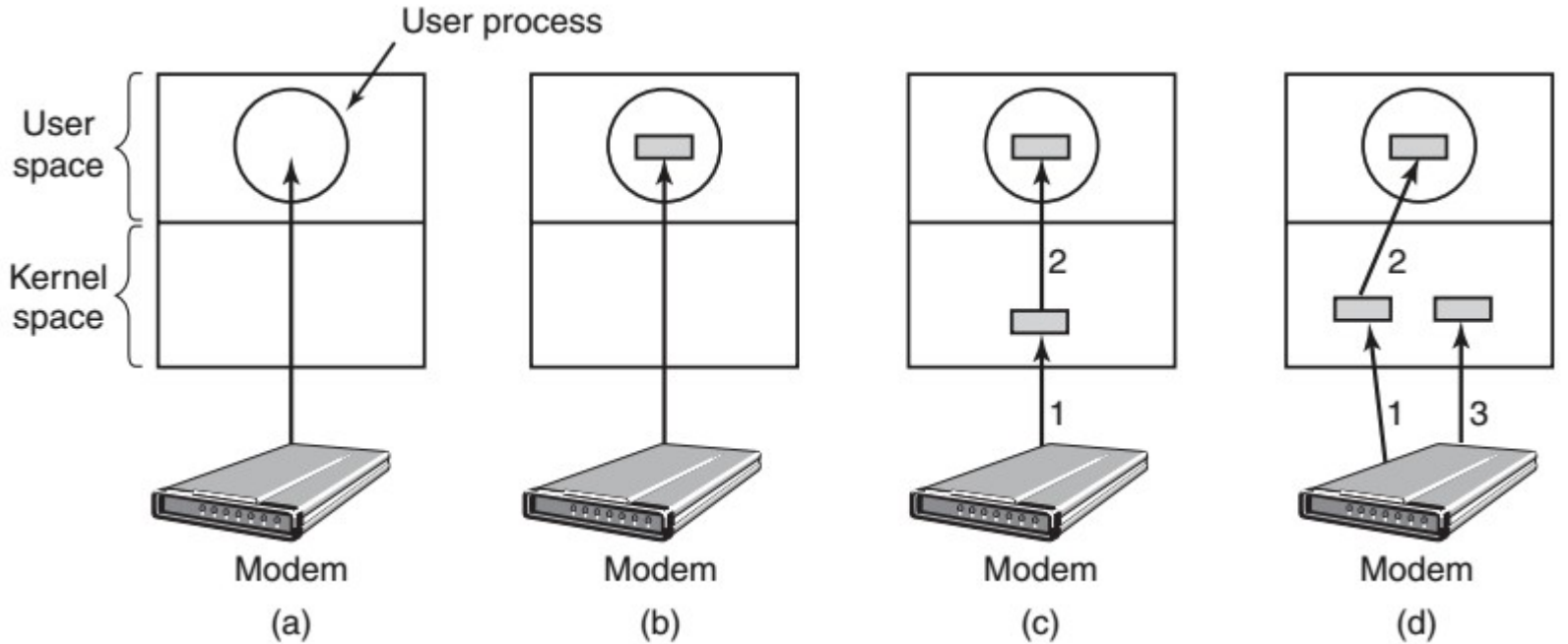
Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA
MAUÁ

- **Software de E/S independente de dispositivo**
  - **Buferização**
    - A buferização **acelera** o **processo** de **ler**/**escrever** de/para um **dispositivo**;
    - A **não buferização implica** em **tratar** uma **interrupção** para cada **unidade** de dado;
    - Pode-se pensar em **utilizar** um **buffer** no **espaço** do **usuário** e, a cada **interrupção** ele **recebe novos dados** – mas por estar **separado** do **kernel**, nada **garante** que ele **esteja** sempre **disponível** e para garantir gasta-se muitos recursos;
    - Uma **alternativa melhor** é manter o **buffer** no **kernel** e melhor, utilizar uma **dupla buferização**: quando um buffer está cheio ele é disponibilizado ao usuário e enquanto isso o outro buffer é utilizado pelo driver;
    - Uma implementação eficiente de buffer é o **buffer circular**.

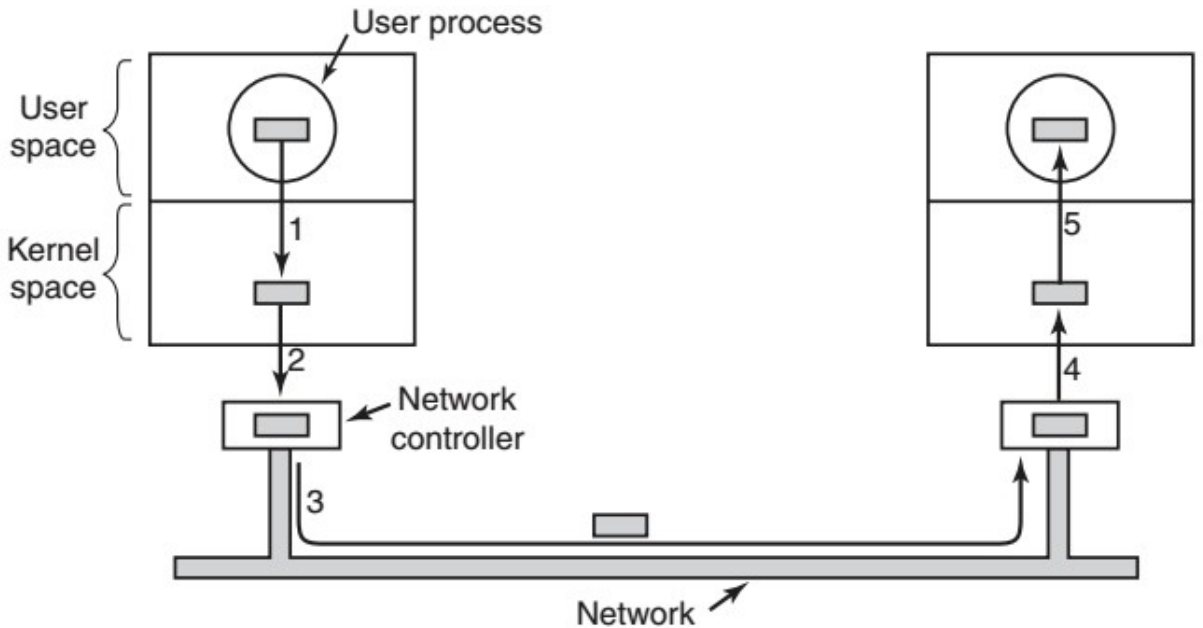
Camadas de software de E/S

- **Software de E/S independente de dispositivo**
  - **Buferização**



Camadas de software de E/S

- **Software de E/S independente de dispositivo**
  - **Buferização**
    - Exemplo onde um dado é copiado diversas vezes



Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA

- **Software de E/S independente de dispositivo**
  - **Reportagem de erros**
    - Erros de programação: escrever para um dispositivo de entrada etc.
    - Uso de parâmetros inválidos
    - Erros de E/S: utilizar um disco rígido com defeito; ler um “bad block” etc.
  - **Alocação e liberação de dispositivos dedicados**
    - Utilizar uma chamada uniforme para alocar o dispositivo – por exemplo, open(), e com seu resultado utilizá-lo ou não.
  - **Tamanho de bloco independente de dispositivo**
    - Utilizar tamanhos de bloco lógicos, independentes de tamanhos de setores físicos.

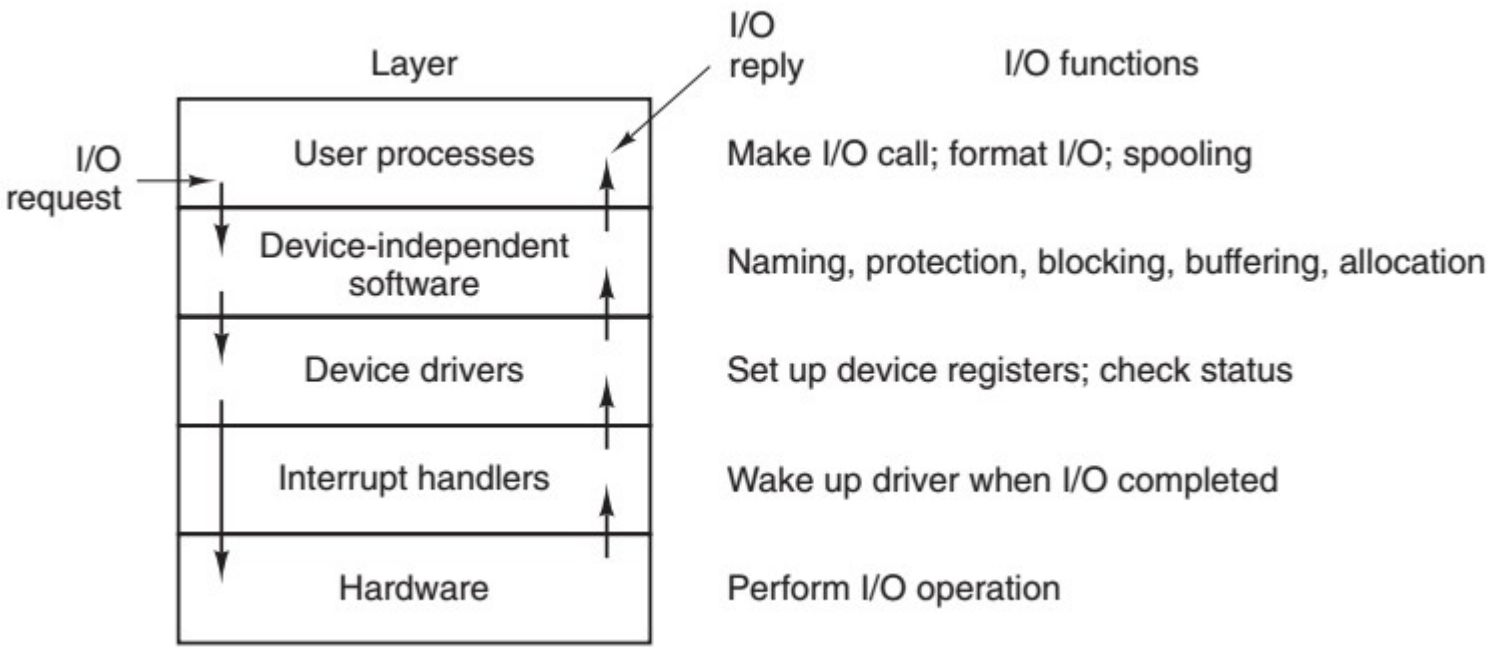
Camadas de software de E/S

INSTITUTO MAUÁ DE TECNOLOGIA

- **Software de E/S no espaço de usuário**
  - Utilizam-se de **bibliotecas** que são **compiladas** aos **programas** de usuário.
  - **Exemplo**: http://man7.org/tlpi/code/online/diff/fileio/copy.c.html

Camadas de software de E/S

- **Resumo**



Referências bibliográficas

TANENBAUM, Andrew S. **Sistemas operacionais modernos**. 3. ed. São Paulo: Pearson, 2013. 653 p.