Arquitetura de Sistemas Operacionais - RESUMO PROVA 1

Sumário

Unidade 1

O que é um sistema operacional (S.O.)?

- É um software ou um conjunto de softwares que cria uma interface entre outros softwares e o hardware no computador, gerenciando seus recursos e seu uso. Ele concede ou nega solicitações de acesso a recursos por parte das aplicações, tomando cuidado ao alocar os recursos, evitando que os programas não interfiram uns com os outros. Em resumo, ele esconde os processos internos do hardware para que os programas se preocupem somente com a lógica do negócio.

Qual o objetivo do S.O.?

- O objetivo é fornecer uma abstração do hardware, tornando transparente para as aplicações de usuário o tipo e as características do hardware disponível. Ele deve lidar com uma série de recursos e para cada um aplicar técnicas de gerenciamento específicas.

Onde fica localizado o S.O.?

- Fica localizado na camada que fica entre o hardware e as aplicações que rodam no seu computador.

Como uma aplicação é executada?

- Uma aplicação primeiro fica guardado no disco e para ser utilizada, vai pra memória RAM, mas não podem interferir nos dados dos outros aplicativos. E tudo isso depende do S.O., ou seja, são serviços fornecidos pelo S.O.

Cite exemplos de serviços fornecidos pelo S.O.

- Enviar dados pela rede, gravar no disco, fornecer memória e conceder acesso a leitura de um arquivo no disco.

Quais são as responsabilidades de um S.O.?

- Gerenciar processos, memória e os dispositivos de entrada e saída, prover um sistema de arquivos, controlar a segurança, prover um mecanismo de comunicação e prover uma interface com o usuário.

Do que se trata o gerenciamento de processos?

- O principal recurso a ser gerenciado pelo S.O. é a CPU (processador), com isso, ele gerencia os processos (programas de usuário em execução), fornecendo serviços para criar, destruir e alterar a prioridade de processos, a comunicação entre os processos e a sincronização entre os processos concorrentes.

Qual a diferença entre escalonamento de processos e troca de contexto?

- Ambos são de responsabilidade do S.O., sendo o escalonamento de processos a escolha de qual programa entrará e execução e a troca de contexto a escolha de qual processo deixará de ser executado.

Do que se trata o gerenciamento de memória?

- O gerenciamento de memória tem a ver com os pedidos de alocação e liberação de memória feito pelos processos (programa de usuário) e o S.O. deve assegurar que os processos não interfiram uns nos outros e que não haja desperdício de memória.

Por que uma aplicação precisa alocar memória?

- Porque ele precisa rodar as instruções que a CPU vai executar e as vezes temos dados que ainda estão sendo trabalhados, sendo assim ele fica na memória e posteriormente podem ser gravados no disco.

Quais os serviços típicos oferecidos pelo gerenciador de memória?

- Solicitar memória adicional diretamente, solicitar memória indiretamente (quando se cria um novo processo), liberar memória (devolvê-la ao S.O.) e solicitar áreas de memória para serem compartilhadas por mais de um processo.

Do que se trata o gerenciamento dos dispositivos de entrada e saída (E/S)?

- Se trata do S.O. “escondendo” os detalhes do hardware, ou seja, ele fornece serviços como abrir um dispositivo e conectá-lo a um processo, ler e gravar dados de/para um dispositivo, fechar e liberar um dispositivo e fornecer acesso exclusivo a um dispositivo.

Do que se trata um sistema de arquivos?

- Como a maioria das aplicações necessitam armazenar dados de maneira persistente, o S.O. fica responsável por prover a interface para o armazenamento e recuperação dos dados. Em resumo: “A aplicação só conhece a palavra arquivo e quer gravar e ler, não importa onde e como é guardado”.

Quais os serviços oferecidos por um sistema de arquivos?

- Abrir, fechar, ler e modificar um arquivo, gravar dados em um arquivo e procurar por um lugar aleatório dentro de um arquivo.

Como funciona a segurança do sistema computacional?

- O S.O. provém mecanismos de segurança para evitar que um processo encerre a execução de outro e que processos acessem dispositivos de E/S sem permissão para tal. E também serve de porteiro, autorizando ou não que um processo acesse qualquer um de seus subsistemas.

Como funciona o mecanismo de comunicação (rede)?

- O S.O. provém um serviço de protocolos para permitir que as aplicações se comuniquem, permitindo estabelecer uma conexão com um serviço remoto, atender conexões de um cliente remoto, enviar e receber mensagens para/de um sistema remoto e fechar uma conexão com um sistema remoto. “A aplicação não precisa saber se o dispositivo está usando cabo de rede, 3g, 4g ou Wi-Fi, ele só precisa estar conectado, e o S.O. fornece esse serviço”.

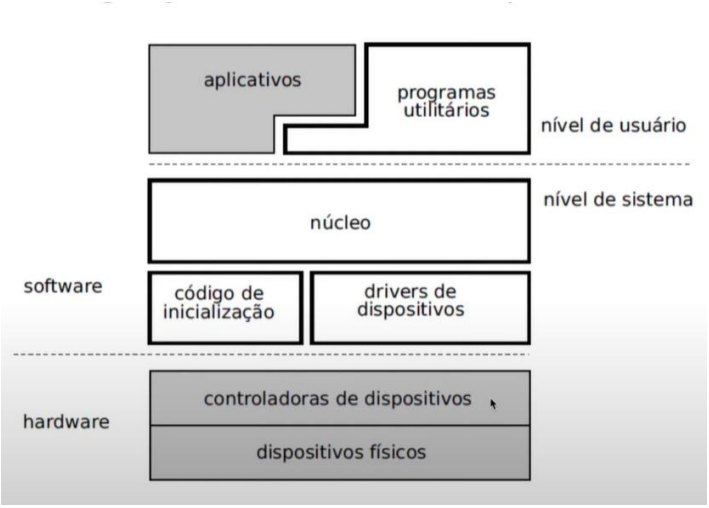
Como funciona a interface com o usuário?

- O S.O. fornece uma interface amigável com o usuário, como um botão, uma janela, etc.

Quais os componentes mais comuns presentes em um S.O. e suas características?

- Kernel (núcleo, onde estão os gerenciadores), drivers (ponte entre o hardware e o S.O. - software disponibilizado pelo fabricante do hardware para fazer a comunicação entre o hardware e o software), código de inicialização (código executado ao ligar o computador e pegar parte do S.O.) e programas utilitários (calculadora etc.).

Um desenho sobre a arquitetura interna do S.O.:

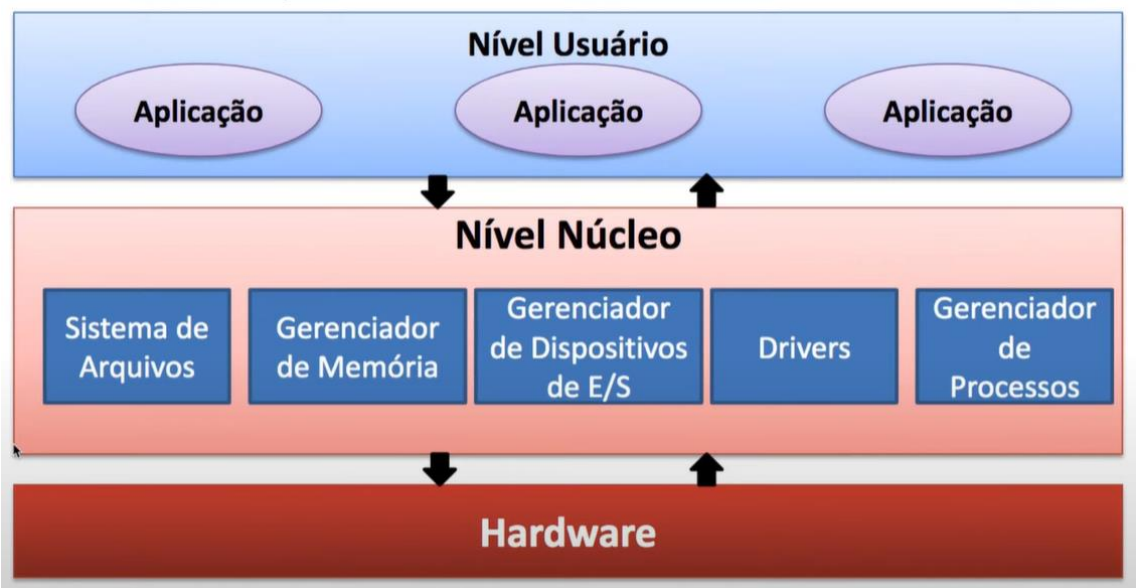


Como um S.O. pode ser organizado?

- Como monolítico, em camadas, micronúcleos e máquinas virtuais.

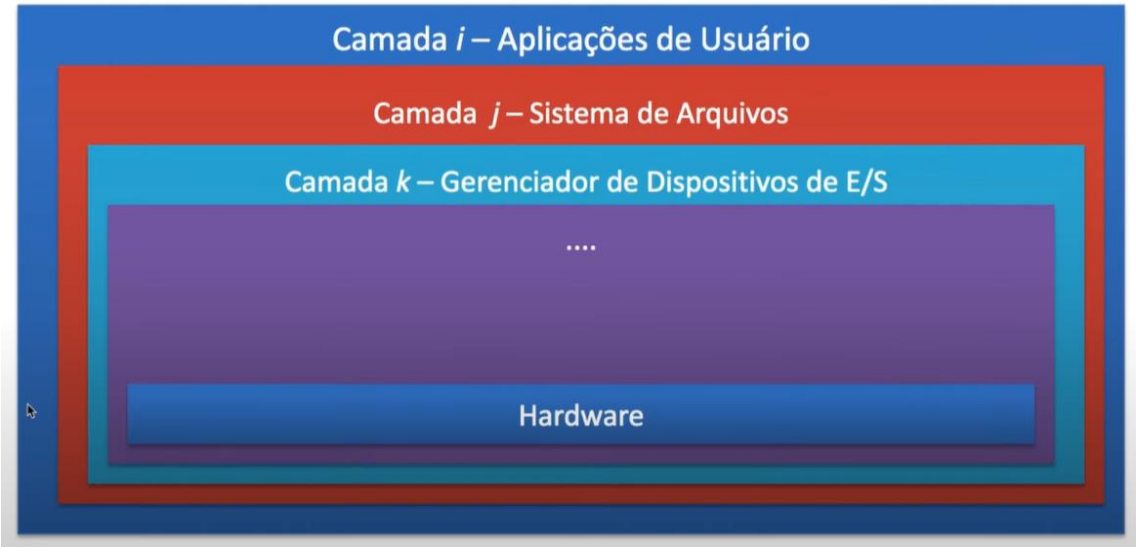
Explique o S.O. monolítico.

- No modo monolítico, o S.O. tem partes dentro dele, mas está colocado tudo como um bloco só, como segue a imagem abaixo. O modo monolítico é mais compacto, possui melhor desempenho, pois todas as partes dele estão dentro de um mesmo bloco e eles se comunicam via memória rapidamente, porém se uma dessas partes quebrar, todas as outras também podem quebrar por estarem no mesmo bloco. Outra desvantagem seria na manutenção, por estar tudo muito acoplado.



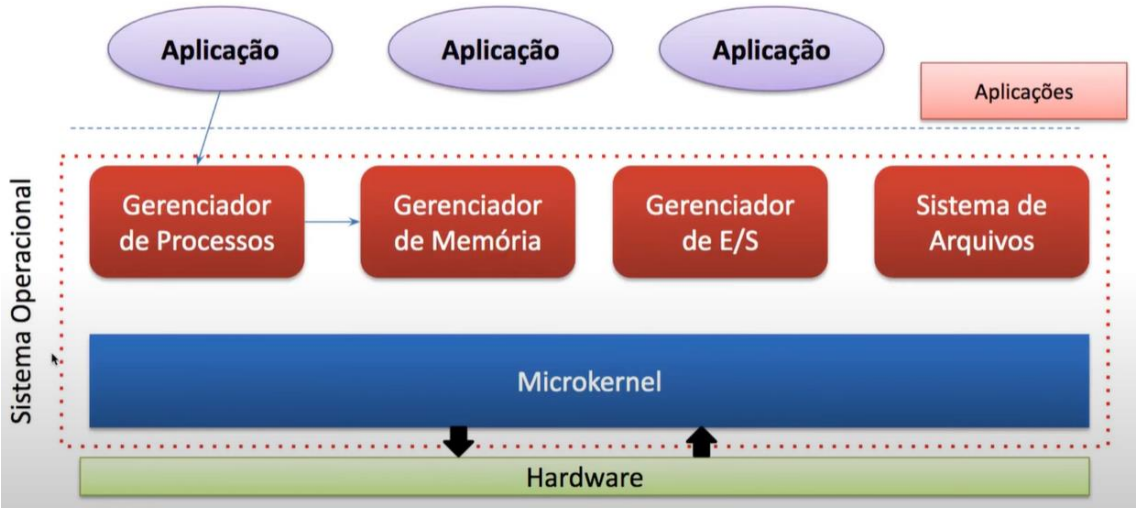
Explique o S.O. em camadas.

- É o modo onde cada camada oferece serviços para a camada de cima, evitando que um problema transite entre as camadas. Sendo a camada de mais alto nível controla as aplicações de usuário e a de mais baixo nível controla o hardware. Sendo assim, cada camada é especializada em uma determinada função do S.O., porém acaba tendo uma perda de desempenho devido a troca de comunicação entre as camadas do sistema. Como o exemplo:



Explique o S.O. em micronúcleo.

Nesse modo, somente os códigos de baixo nível permanecem no núcleo, o restante é implementado fora do núcleo (na camada de usuário). A comunicação é feita através de troca de mensagens, porém ela é mais devagar por conta disso, entretanto se um módulo falhar, o erro não se alastra pelo sistema. Segue a imagem do modelo:

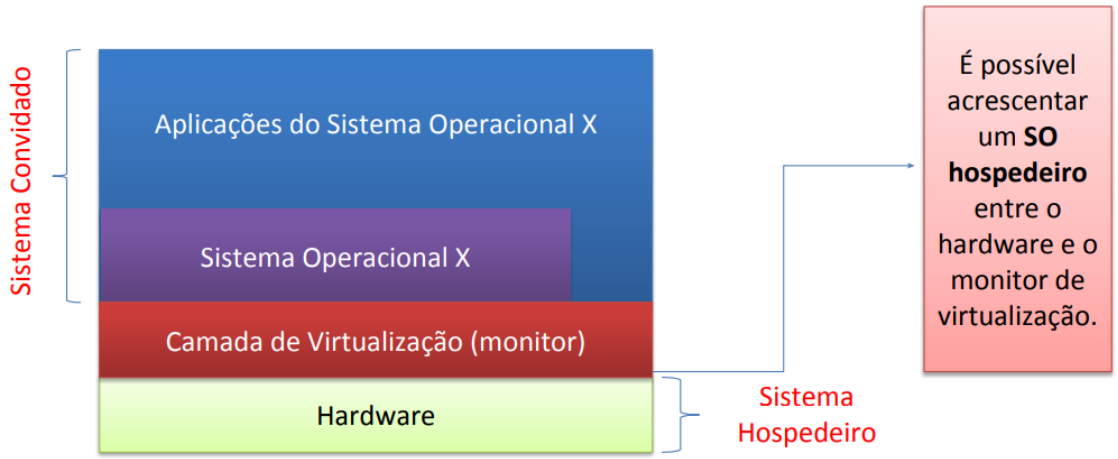


Quais são os modelos do S.O. máquina virtual?

- Máquina virtual de aplicação, onde é destinada a suportar apenas um processo ou aplicação específica (como a máquina virtual Java, JVM). Máquina virtual de sistema, onde o S.O. vai usar a máquina virtual achando que está utilizando o hardware (como VirtualBox e VMWare).

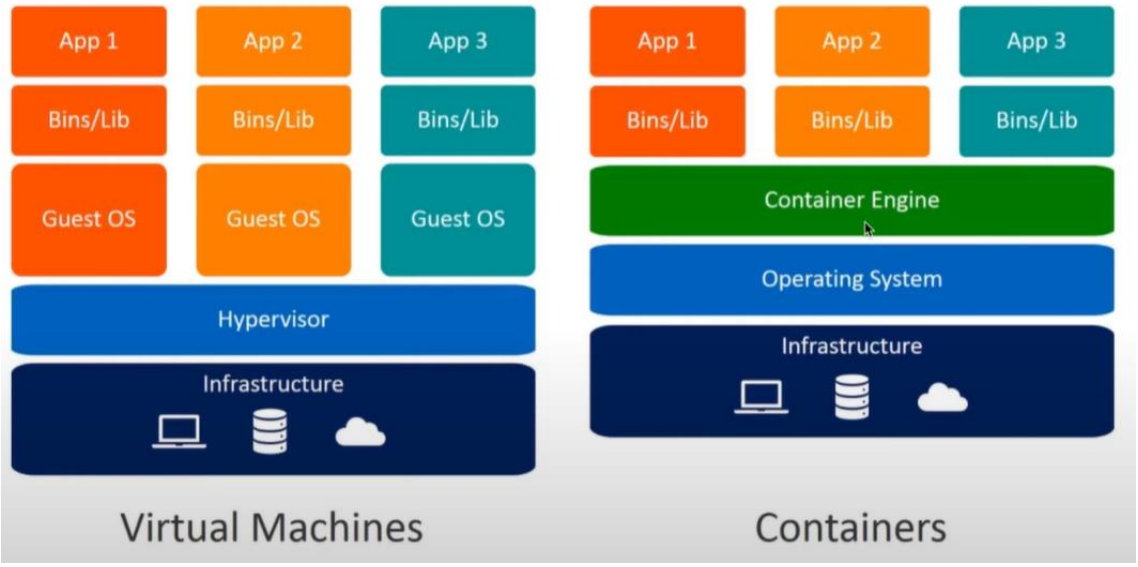
Quais os componentes da máquina virtual de sistema?

- Um sistema real (hospedeiro), que vai ter acesso real ao hardware, uma camada de virtualização, para que em cima dela eu possa rodar o S.O. convidado e um sistema convidado que executa sobre o sistema virtualizado. Como a imagem abaixo:



Explique a máquina virtual de sistema usando contêineres.

- É a mesma ideia da máquina virtual de sistema normal, porém os usuários compartilham o mesmo S.O., mas cada um olhando apenas para o seu. Segue a imagem exemplo entre a diferença:



Quais são os tipos de S.O.?

- Eles são classificados de acordo com o conjunto de serviços oferecidos e a aplicação ao qual se destina. E são classificados em batch (lote), rede, distribuído, multiusuário desktop, servidor, embarcado e tempo real.

O que é um S.O. do tipo Batch (lote)?

- Ele não tem interação com o usuário e se caracteriza por organizar uma fila de todos os programas a serem executados.

O que é um S.O. do tipo Rede?

- É um S.O. que tem acesso a rede, possuindo suporte para operações nas mesmas, como por exemplo Windows, Android, Linux, etc.

O que é um S.O. do tipo distribuído?

- É um S.O. onde as partes do seu sistema rodam em máquinas separadas, de forma transparente ao usuário.

O que é um S.O. do tipo multiusuário?

- É um S.O. que permite que vários usuários acessem os recursos disponíveis e ao mesmo tempo, mas vale destacar que é importante possuir um método de autenticação dos usuários.

O que é um S.O. do tipo desktop?

- É um S.O. onde utilizamos em computadores e notebooks, como Windows, Linux, MacOS, etc.

O que é um S.O. do tipo servidor?

- É um S.O. dedicado a hospedar sites e serviços, projetado para permitir a gestão de eficiente de grande quantidade de recursos.

O que é um S.O. do tipo embarcado?

- É um S.O. projetado para executar e gerenciar hardware com poucos recursos computacionais ou hardwares dedicados a uma função específica, como o hardware pra controlar os freios abs de um carro.

O que é um S.O. do tipo tempo real?

- É um S.O. projetados para atender prazos, como por exemplo controlar o tempo de uma cancela fechar para o trem passar. E podem ser de dois tipos, soft real time (pode perder um prazo ou outro, como software de videoconferência) e hard real time (críticos que não podem perder nenhum prazo).

O que são as chamadas de sistema?

- São um método de acessar os serviços que o sistema operacional oferece, como se fosse uma API que o S.O. permite utilizar na hora do desenvolvimento de aplicações que vão rodar em cima do sistema operacional.

Cite exemplos de chamadas de sistema e explique.

- fork(), que cria um novo processo fazendo uma cópia do processo existente, exit(), que encerra o processo solicitante, open(), que abre um arquivo para leitura/escrita, read(), que lê dados de um arquivo aberto e write(), que escreve dados em um arquivo aberto.

Descreva a evolução do S.O. ao longo das décadas.

- Entre as décadas de 40 e 50 não havia um S.O., então o desenvolvedor de cada aplicação tinha que se preocupar com todo o funcionamento do hardware para cada aplicação.

Nas décadas de 60 e 70, surge o conceito de tempo compartilhado e inicia alguns S.O., já nas décadas de 80 e 90, foi desenvolvido os primeiros S.O. para desktop e a partir de 2000 os S.O. foram melhorados e surgiram os primeiros S.O. para telefones.

Quem fica entre as aplicações e o hardware?

- O Sistema Operacional.

Questionário 1

Interface de programação que expõe funcionalidades do sistema operacional aos desenvolvedores de aplicações. → **Chamadas de sistema.**

Aplicações executam sobre um software que emula os recursos de um sistema operacional → **Virtualização por contêineres.**

Componentes do sistema operacional são processos distintos que se comunicam. → **Microkernel.**

Software que permite inicializar o sistema operacional → **Código de inicialização**.

Foco em executar tarefas dentro dos prazos estabelecidos → **Sistema operacional do tipo tempo real.**

Responsável por gerir recursos de hardware → **Sistema operacional.**

Componentes do sistema operacional agrupados em uma única entidade de execução. → **Kernel monolítico.**

Software de interface entre sistema operacional e controlador de dispositivos de hardware. → **Driver.**

Foco em executar o máximo de tarefas por unidade de tempo → **Sistema operacional do tipo lote.**

Sistema operacional executa sobre um software que emula os recursos de hardware → **Monitor de virtualização**.

Unidade 2

As aplicações precisam armazenar e recuperar informações? Explique.

- Sim, pois há momentos que os dados devem persistir por mais tempo do que o tempo de vida de um processo, como um bando de dados, onde é inaceitável que a informação em uso pelo processo desapareça quando ele é encerrado. Porém a capacidade de armazenamento está restrita ao tamanho do espaço de endereçamento virtual.

Quais as questões a serem gerenciadas pelo S.O.?

- Como encontrar uma informação, como impedir que um usuário tenha acesso a informações de outro usuário e como saber quais blocos estão livres.

O que é um arquivo?

- Arquivo é uma unidade lógica de informação que pode ser criada, alterada e removida por um processo.

O que é a persistência de um arquivo?

- É quando a informação armazenada não é deletada quando um processo se encerra e não pode ser afetada pela criação ou pelo término de um processo, ou seja, todo arquivo é persistente.

Como os arquivos são definidos pelo S.O.?

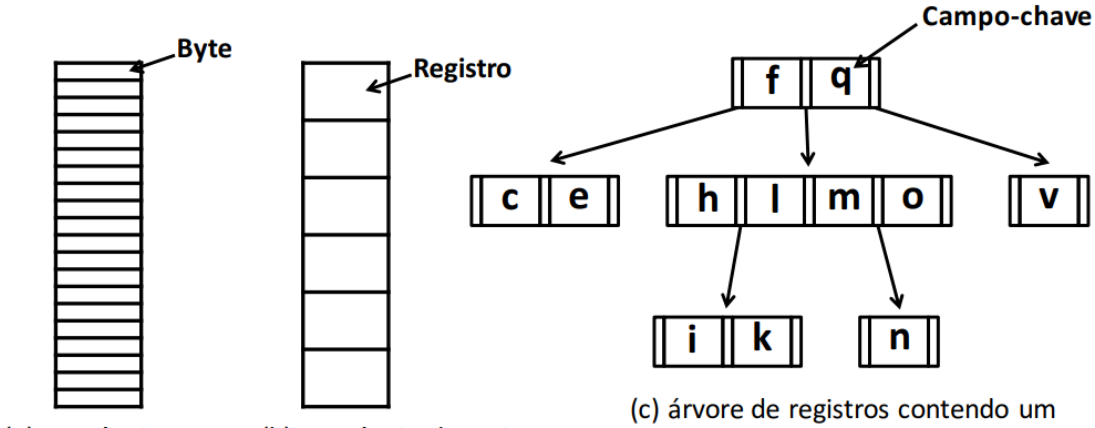
- Como nomeados, estruturados, acessados, protegidos e implementados.

O que é um sistema de arquivos?

- É a parte do S.O. que trata dos arquivos.

Quais as estruturas dos arquivos?

- Sequência desestruturada de bytes, sequência de registros de tamanho fixo e árvore de registros contendo um campo chave em posição fixa no registro (árvore ordenada pela chave). Segue o exemplo abaixo:



(a) sequência desestruturada de bytes

(b) sequência de registros de tamanho fixo

(c) árvore de registros contendo um campo chave em uma posição fixa no registro (árvore ordenada pela chave)

Explique os tipos de arquivos existentes.

- Arquivos regulares, que são os arquivos de usuário, como um arquivo pdf, e os diretórios, que são arquivos do sistema que mantém a estrutura do sistema de arquivos.

Explique os tipos de arquivos regulares.

- Arquivos de texto, onde o arquivo em binário pode ser traduzido para caracteres e arquivos binários, que não podem, como por exemplo um arquivo executável.

Qual a diferença entre arquivo de acesso sequencial e arquivo de acesso aleatório?

- Em arquivos de acesso sequencial, onde você para acessar byte n, deve ler do byte 0 até o byte n, ou seja, de forma sequencial, e isso ocorre, por exemplo, se o arquivo estiver guardado numa fita magnética. E em arquivos de acesso aleatório, os bytes podem ser lidos em qualquer ordem, ou seja, para ler o byte x, não precisa percorrer do 0 até o x, pode ler o x direto, isso ocorre, por exemplo, se ele estiver gravado no disco.

O que são atributos de arquivos?

- São os metadados, que armazenam seus nomes e dados, entre outras informações como tamanho atual, criador, senha (se houver).

Como funcionam as operações com arquivos?

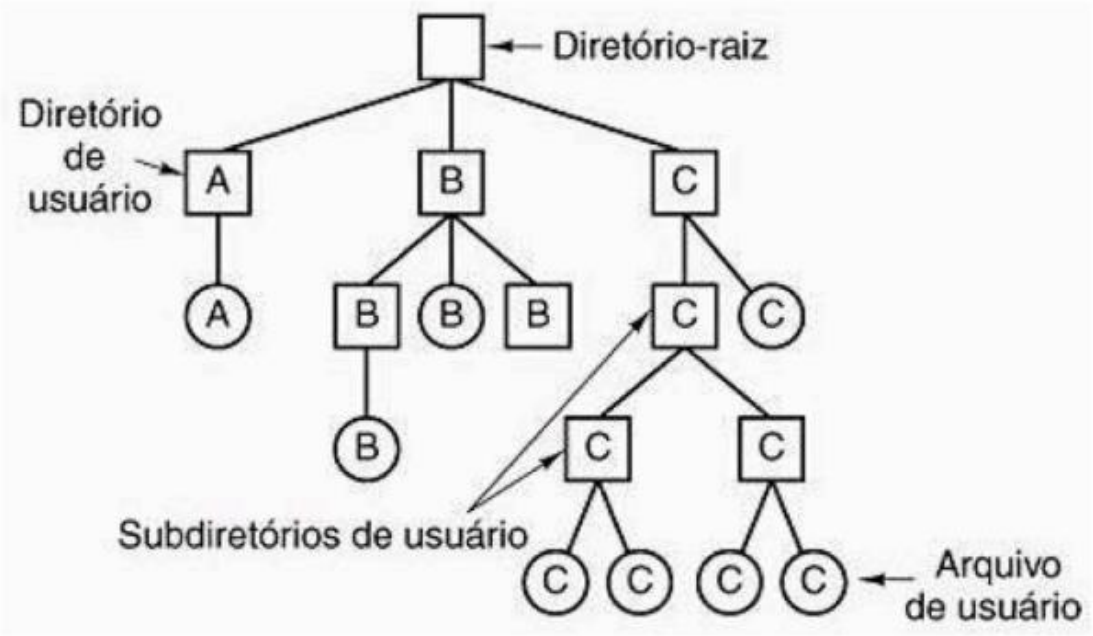
- Normalmente a manipulação de arquivos é feita estritamente através de chamadas de sistema, isso garante, por exemplo, que um usuário não possa modificar um arquivo de outro usuário.

Explique cada uma das operações com arquivos.

- Create: cria um arquivo vazio; Delete: deleta um arquivo;
- Open: abre um arquivo já existente e ao abrir retorna um descritor e o mapeia na memória RAM;
- Close: fecha o arquivo, liberando espaço na RAM;
- Seek: reposiciona o ponteiro para um local específico do arquivo, podendo ir p esquerda ou direita desse arquivo;
- Read: pode ler byes a partir desse ponteiro para armazenar esses dados lidos;
- Write: guarda em memória ou salva dados de buffer em um arquivo;
- Append: adiciona dados ao final do arquivo;
- Get attributes: retorna os atributos do arquivo;
- Set atributes: modifica os atributos do arquivo;
- Rename: altera o nome do arquivo;

O que são diretórios?

São um tipo especial de arquivo, onde são listados os arquivos ou outros diretórios que estão dentro do diretório, organizando, assim o sistema de arquivos. E suas operações ocorrem da mesma maneira que os arquivos, porém com algumas operações a mais, como Link (criar um "atalho" desse diretório em outro diretório). Segue um print sobre a hierarquia dos diretórios:



Qual a diferença entre caminho absoluto e caminho relativo?

- O caminho absoluto sempre começa na raiz, como /usr/lib/dict, já o relativo é a partir do diretório atual, como ./dict caso esteja no diretório atual ou ../dict para caso esteja no diretório pai.

Como é feita a utilização de um arquivo?

- Para utilizar um arquivo é necessário associá-lo a um stream e, então, manipular a stream.

Quais os tipos de Stream?

- Stream de texto, onde se lê caracteres e Stream binária, onde se lê bytes.

Como é feita a leitura de arquivos em disco?

- Ao abrir um arquivo, o indicador de posição aponta para o começo do arquivo e à medida que cada caractere ou byte é lido ou escrito no arquivo, o indicador de posição é incrementado.

O que é flushing?

- É quando é chamada a função close enquanto o conteúdo ainda está sendo escrito, porém é garantido que o conteúdo seja escrito no dispositivo externo, e garantindo que nenhuma informação seja acidentalmente deixada no buffer de disco.

Como se manipula um arquivo em C?

- Primeiro se faz um ponteiro com FILE, depois abre esse arquivo (através de vários modos, principalmente com o r+), pode fechar o stream, pode mover o ponteiro do próximo ponteiro a ser lido (fseek), pode remover, renomear e esvaziar o conteúdo do stream, como também pode escrever nesse arquivo (pode salvar até uma struct). Já um arquivo binário é diferente, pois é lido em bytes. Ao ler um arquivo, não se sabe do que se trata, então tudo é lido em bytes.

O que é um buffer?

- São os dados (ponteiro) a serem escritos em um arquivo.

O que é um controle de acesso?

- É a solução criada para o impedir que usuários não autorizados leiam ou alterem dados. Resolvendo assim um problema na segurança.

Quais os requisitos do controle de acesso?

- Distinguir o dono do arquivo/diretório entre os usuários e permitir que o dono defina quem pode acessar seus arquivos e qual tipo de acesso permitir.

Qual a diferença entre acesso de baixo nível e acesso de alto nível?

- No de baixo nível é permitido ler, escrever e executar um arquivo e no de alto nível é permitido deletar (pode exigir escrever) e copiar (depende de ler).

O que é uma lista de controle de acesso (ACL)?

- É a listagem de uma tabela onde relaciono arquivos, usuários e os tipos de acesso.

Quais a vantagens e desvantagens de ACLs?

- Ela é muito expressiva e detalhada, com granularidade pequena, sendo possível detalhar o mínimo possível sobre o arquivo, usuário e o tipo de acesso, porém é exaustivo manter a manutenção e é propensa a erros, além de ter tamanho variado, tendo dificuldade de armazenar a ACL em diretórios.

O que é a simplificação de ACLs?

- É o aumento da granularidade (reduzir expressividade), onde para cada arquivo se identificam o nodo, grupo (que compartilha o arquivo), outros (quem não faz parte do acesso) e os modos de acesso (read, write, execute).

Exemplo de ACLs simplificada em sistemas Unix:

1 = acesso permitido, 0 caso contrário

Usuário	Bit Read	Bit Write	Bit eXecute	RWX Decimal
Dono	1	1	1	7
Grupo	1	0	1	5
Outros	1	0	0	4

Comando: **chmod 754 nome_do_arquivo**

7, 5 e 4 são os números decimais correspondentes ao binário formado com os 0 e 1.

O que são e quais são os setores da implementação?

- Setor é a maneira a qual o disco é dividido e ele divide-se em header, dados e trail, onde o header e o trail permitem verificar a integridade de dados.

Onde os setores são agrupados?

- São agrupados sequencialmente em blocos que geralmente são de 512 bytes.

Em qual setor do disco está armazenado o código para iniciar o sistema (MBR)?

- O chamado Registro Mestre de Inicialização (MBR) está armazenado no setor 0 do disco.

O que acontece quando o computador é ligado?

- A BIOS lê e executa o MBR, logo o programa do MBR localiza a partição ativa, lê seu primeiro bloco (bloco de inicialização) e o executa e, em seguida, o programa do bloco de inicialização carrega o S.O. contido naquela partição.

Explique as partições do disco.

- Bloco de inicialização: onde vai carregar o S.O.;
- Superbloco: onde armazena informações sobre o tipo de sistema de arquivos e o número de blocos;

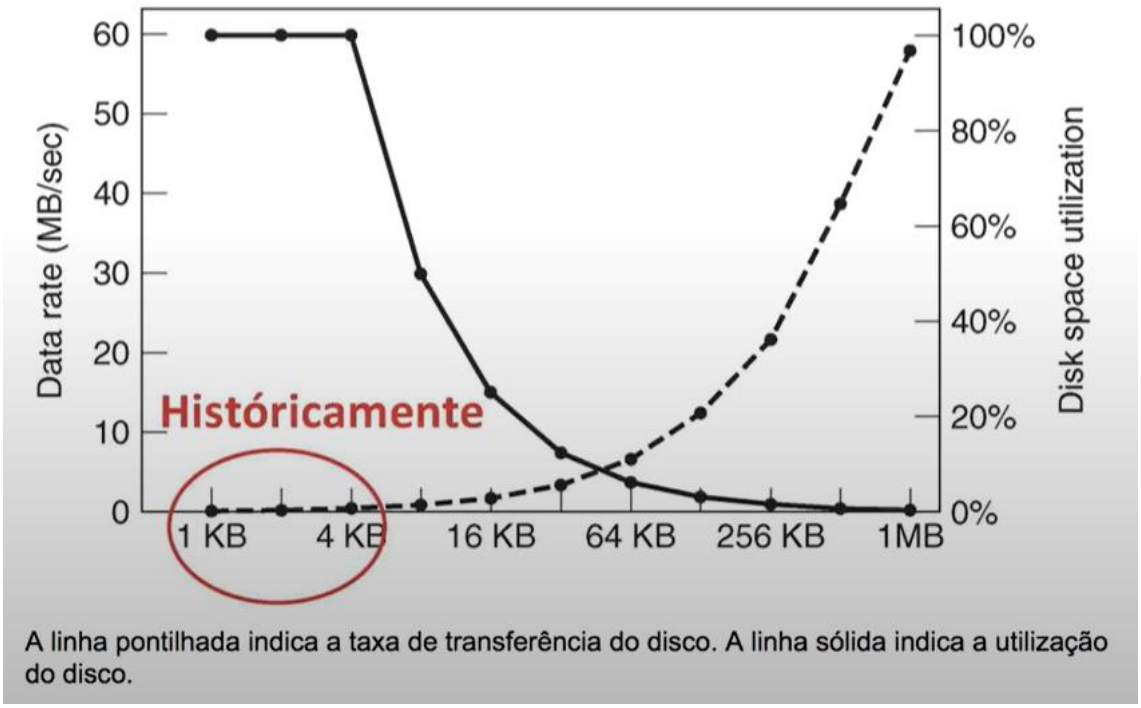
- Gerenciamento de espaço livre: lista de blocos livres no disco;
- I-nodes: estruturas que armazenam as informações sobre cada arquivo e onde localiza-los;
- Diretório-raiz: armazena o topo da árvore do sistema de arquivos;
- Arquivos e diretórios: armazena os arquivos e diretórios propriamente ditos.

Quais as duas estratégias gerais do gerenciamento do espaço em disco?

- Alocar n bytes consecutivos de espaço em disco e dividir os n bytes em diversos blocos não necessariamente contíguos.

Quais as vantagens e desvantagens de cada estratégia do gerenciamento do espaço em disco?

- Ao alocar n bytes consecutivos de espaço em disco, é necessário mover os dados quando o arquivo cresce e não há espaço contíguo e tem um custo mais alto, pois o disco é muito mais lento do que a memória RAM, com isso, quase todos os sistemas de arquivos quebram os arquivos em blocos de tamanho fixo. E ao dividir os n bytes em diversos blocos não necessariamente contíguos, tem a questão de saber o tamanho do bloco e haverá um balanço entre bloco grande (desperdício de disco) e bloco pequeno (ocuparão muitos blocos no disco, com múltiplas buscar, reduzindo o desempenho). Segue um exemplo do segundo caso:



A linha pontilhada indica a taxa de transferência do disco. A linha sólida indica a utilização do disco.

Quais os diferentes métodos podem ser usados para relacionar os blocos do disco com os arquivos? Explique-os.

- Alocação contígua: consecutivos, rápido, mas fragmenta o disco;
- Alocação por lista encadeada: um bloco do arquivo A aponta pro bloco do arquivo B e assim vai, assim não fragmenta o disco, porém o acesso aleatório é lendo, pois teria que ler todos os blocos anteriores e também parte do bloco usada p armazenar o próximo ponteiro, não pode ser usada para armazenar dados;
- Alocação por lista encadeada usando uma tabela na memória: armazena as informações dos ponteiros da lista encadeada em uma tabela em memória, tabela chamada de Tabela de Alocação de Arquivos (FAT), porém essas tabelas ficam muito grandes, sendo muito desvantagem pois vai ocupar muito espaço na RAM;
- i-nodes: vai relacionar pra cada arquivo, os atributos e os endereços dos blocos do arquivo, assim, ao invés de ter os endereços salvos nos blocos, ele salva todos os endereços de bloco em um único bloco. E os i-nodes são divididos em vários níveis.

Explique os níveis dos i-nodes.

- Primeiro nível: Os 3 primeiros índices armazenam atributos do arquivo, como modo de acesso, leitura e escrita e depois os primeiros endereços de blocos do arquivo são armazenados diretamente no i-node e também se tem os blocos de endereços (3 últimos);
- Segundo nível: Chamado de bloco indireto simples, é usado caso não haja espaço no i-node para todos os blocos, onde ele armazena o endereço dos blocos que não couberam no i-node. É o antepenúltimo índice do i-node;
- Terceiro nível: Chamado de bloco indireto duplo, é usado caso o segundo nível não tenha sido suficiente, que é um bloco que aponta para blocos indiretos simples. É o penúltimo índice do i-node.
- Quarto nível: Chamado de bloco indireto triplo, é o endereço para um bloco de disco que contem uma lista de blocos indiretos duplos e é o último índice do i-node.

Quais as vantagens da implementação de arquivos?

Permitir lidar com arquivos muito pequenos ou muito grandes de forma eficiente e acesso rápido a arquivos pequenos, pois os mesmos possuem seus endereços armazenados diretamente no i-node.

Qual a função do sistema de diretórios?

- A função dele é mapear o nome do arquivo (ou caminho) na informação necessária para localizar os seus dados. E essa informação depende do método de alocação de blocos no disco, se for contígua, retorna o endereço de disco onde começa o arquivo, se for lista encadeada, retorna o numero do primeiro bloco e se for i=node, retorna o número do i-node.

Quais as duas maneiras de monitorar os blocos livres?

Através da lista encadeada de blocos livres ou pelo mapa de bits.

O que é uma lista encadeada de blocos livres?

- É uma maneira de monitorar os blocos livres, sendo um bloco que possui um endereço de blocos livres e um endereço (último) para o próximo elemento da lista encadeada.

O que é o mapa de bits?

- É uma maneira de monitorar os blocos livres, onde cada bit representa um bloco no sistema de arquivos, onde um bit com 0 está livre e um bloco alocado tem 1. Possui a vantagem de ser eficiente e a desvantagem de consumir muita memória RAM.

O que é o Network File System (NFS)?

- É um sistema de arquivos baseado em rede, que serve para compartilhar arquivos entre computadores remotos, seu compartilhamento é baseado na arquitetura cliente-servidor, onde o servidor disponibiliza o diretório a ser compartilhado e o cliente acessa remotamente o diretório compartilhado para ler ou escrever. E ainda abstrai as diferenças de sistemas de arquivos locais entre cliente e servidor, ou seja, posso ter um servidor numa máquina Linux e um cliente em uma máquina Windows.

Explique o protocolo NFS.

- O protocolo NFS provê um conjunto de operações remotas (RPC), como buscar um arquivo em um diretório, ler um conjunto de entradas em um diretório, manipular links e diretórios, acessar atributos de arquivos e ler e escrever arquivos, porém essas operações são permitidas somente quando o diretório remoto estiver montado no sistema de arquivos local do cliente.

Explique o Virtual File System (VFS).

- Muito parecida com a ideia de memoria virtual, onde o sistema operacional abstrai a memoria física, o VFS permite acessar arquivos locais e remotos da mesma forma.

Quais as duas formas de melhorar o desempenho? Explique-as.

- Cache de blocos (buffer cache): mantem na memória, blocos do disco que estão sendo utilizados para melhorar o desempenho, eles são trazidos para a buffer para serem lidos/modificados. E cada buffer na buffer cache contem um número de bloco alocado e o status.
-Leitura antecipada (prefetching): transfere blocos do disco para a buffer cache antes mesmo dos mesmos serem necessários, como por exemplo uma leitura sequencial de arquivos.

Unidade 3

O que é um processo?

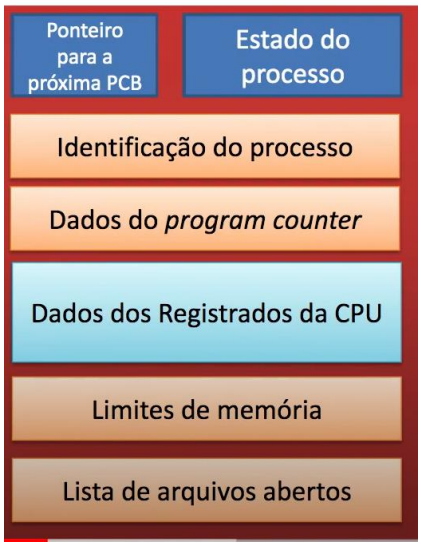
- Processo é uma entidade ativa no sistema e representa um programa em execução. Nota-se que é possível ter mais de um processo para a mesma aplicação.

O que é PCB?

- PCB é a sigla de Process Control Block (bloco de controle do processo) e é uma estrutura de dados que representa um processo no S.O. e possui informações sobre o contexto de execução do processo.

Quais são as informações contidas na PCB?

- Identificação do processo, estado de execução, dados do contador de programas, dados dos registradores da CPU, informações para o escalonamento de CPU, Informações sobre memória, informações de contabilidade e informações sobre as operações de E/S. Segue uma imagem exemplificando-o:



Como é feita a criação de um processo?

- Cada S.O. possui chamadas de sistema responsáveis pela criação e destruição de processos. Existe uma chamada de sistema fork() que cria uma cópia idêntica ao processo que chamou a execução de fork() e assim possuí sua própria PCB. Os possíveis valor de retorno para a função fork() é negativo para erro, positivo para mostrar ao pai a identificação do novo processo criado e 0 para o processo criado (filho).

Quais as chamadas de sistema relacionada a processos (ambiente Unix e Linux)?

- fork(): cria um novo processo;
- exec(): permite que um processo execute um fluxo de execução diferente do fluxo do processo pai;
- exit(): força a destruição do processo;
- getpid(): retorna o ID do processo;
- getppid(): retorna o ID do pai do processo;
- wait(): aguarda a execução do processo filho;
- sleep(): aguarda um tempo em segundos;

O que é multiprogramação?

- É uma técnica onde mantêm-se diferentes processos na memória e intercala-se a CPU entre eles para evitar a CPU de ficar ociosa. Há um caso específico de multiprogramação chamado de multitarefa (time-sharing) para reduzir tempo de resposta para usuários, onde cada processo pode usar a CPU por tanto tempo contínuo e limitado, terminando, assim, os mais curtos primeiro e deixando os mais longos para o final.

O que é troca de contexto e quando ela é realizada?

- É a troca de um processo em execução por outro, assim armazenando os dados de contexto do processo que sairá do processador em sua PCB. Ela é realizada pelo despachante e a escolha de qual processo deverá assumir o processador é realizada pelo algoritmo de escalonamento de tarefas.

O que são interrupções?

- Eventos que interrompem a CPU, como interrupções de relógio (clock ticks).

Quais os passos para a troca de contexto?

- Tem o tratador de interrupção, o despachante (que salva na memoria tudo do processo que estava executando), dai chama o escalonador, que escolhe o próximo processo a ser executado e logo o despachante prepara (restaura PCB) a CPU para receber o processo novo.

Quais os estados de um processo?

- Novo: acabou de ser criado;
- Pronto: está pronto para ser executado. Está na fila de processos aptos;
- Executando: está em execução;
- Aguardando: está aguardando um evento;
- Concluído: concluiu sua execução.

Segue uma imagem exemplificando os estados de um processo:

Quando a troca de contexto pode ocorrer?

- Quando o tempo de execução do processo atual se esgotou (como no caso do time-sharing), quando ocorrer uma interrupção de hardware ou de software ou quando ocorrer um erro de execução.

Quando escalonar a CPU?

- Quando um processo passa do estado de execução para o estado de espera;
- Quando um processo passa do estado de execução para o estado de pronto;
- Quando um processo passa do estado de espera para o estado de pronto;
- Quando um processo termina.

Quais são os critérios para a escolha de um algoritmo de escalonamento de processos?

- Utilização da CPU: manter a CPU o máximo do tempo ocupada;
- Throughput (vazão): quantidade de processos que são concluídos por unidade de tempo;
- Turnaround: o intervalo de tempo a partir da criação do processo até a sua conclusão;
- Tempo de Espera: é a soma dos períodos gastos em espera na fila de prontos;
- Tempo de Resposta: tempo entre envio de uma solicitação e a primeira resposta ao usuário.

Quais os tipos de escalonadores de processos?

- Tem o preemptivo, que tira a CPU do processo quando o tempo já esgotou ou quando outro processo necessita da CPU e o cooperativo, onde somente libera a CPU quando o processo realiza alguma operação de E/S.

O que é um algoritmo de escalonamento FCFS?

- Usa a estrutura de dados fila (FIFO), logo o primeiro processo que solicita a CPU será o primeiro a ser executado. Ele não é preemptivo e o tempo médio de espera dele pode ser longo. Tem as vantagens de ser simples e todos os processos são executados, porém conta com a desvantagem de que processo longo atrasa os próximos processos.

O que é um algoritmo de escalonamento SJF?

- É a ideia de processo mais curto primeiro, ou seja, atribui prioritariamente os processos que tem o pico de CPU mais curto e quando dois processos ou mais tiverem o mesmo pico de CPU, utiliza-se a FCFS. E ele tem a opção de ser preemptivo, ou seja, caso chegue um processo com tempo de CPU menor, ele vai parar de ser executado dando espaço ao de tempo menor. Possui a vantagem de ter o melhor throughput (vazão maior, maior número de processos por unidade de tempo), porém é difícil prever o tempo de CPU e os processos longos podem nunca ser executados (stravation).

O que é um algoritmo de escalonamento baseado em prioridades?

- É um algoritmo onde cada processo é associado a um número que indica sua prioridade, sendo o número maior uma maior prioridade e assim sendo executado primeiro ou vice-versa, ele pode ser definido de ambas as formas, mas geralmente os números menores são os de maior prioridade. Tem a possibilidade de ser preemptivo ou não preemptivo. Uma desvantagem é que um processo com prioridade menor pode nunca ser executado, o que pode ter como solução o envelhecimento de prioridade (que vai aumentando ao longo do tempo na fila).

Quais as soluções para a inversão de prioridades?

- Utilizar no máximo duas prioridades ou herança de prioridade, onde o que está executando assume o valor da prioridade de quem está aguardando até ser finalizado.

O que é um algoritmo de escalonamento Round-Robin?

- É um algoritmo conhecido como revezamento, ou seja, ele foi projetado para sistemas de tempo compartilhado. Ele é parecido com o FCFS, mas com preempção baseada em um tempo conhecido como quantum (geralmente de 10 a 100 milissegundos). E pode ser implementado como uma fila circular, ou seja, o primeiro processo ao ser interrompido, voltará para o final da fila. Mas deve-se prestar atenção no tempo de Quantum vs. tempo de troca vs. tempo de resposta.

O que é um algoritmo de escalonamento de múltiplas filas?

- É um algoritmo que combina os algoritmos em filas, onde cada fila tem seu próprio algoritmo de escalonamento. Podem ser os processos do sistema, processos dos usuários e processos de baixa prioridade. Ou seja, pode haver uma mescla entre os algoritmos.

O que é um algoritmo de escalonamento em sistemas de tempo real?

- São sistemas que tem que cumprir um limite de tempo (deadline), ou seja, tem que terminar de processar os processos dentro do prazo. São divididos em 2 casos, os críticos e os

não críticos, onde os críticos não podem de jeito nenhum perder um deadline, como um sistema de freio por exemplo, e um não critico seria um sistema de transmissão de áudio e vídeo. E podem ser classificadas como aperiódicas (ou assíncronas), que ocorre quando possui um caráter aleatório, ou seja, aparecem de modo aleatório, ou podem ser classificadas como esporádicas, que também possui caráter aleatório, porem existe um intervalo mínimo conhecido entre duas execuções.

O que é um algoritmo de taxa monotônica (RM)?

- Ele possui uma politica de escalonamento estática com preempção e cada processo recebe uma prioridade inversamente proporcional ao período, ou seja, quanto maior o tempo, menor a prioridade. E considera que o tempo de processamento de um processo periódico é o mesmo em cada pico de CPU.

O RT monotônico é escalonável?

- Aplicando a fórmula abaixo, se der acima de 0.82, significa que não é escalonável.

m = total de eventos periódicos
C = tempo de execução máximo do evento
P = período do evento

$$\sum_{i=1}^{m} \frac{C_i}{P_i} \leq 2(2^{1/m} - 1)$$

0.82...

O que é o algoritmo do Deadline mais cedo primeiro (EDF) num escalonamento em sistemas de tempo real?

- É o algoritmo onde as prioridades são distribuídas dinamicamente, logo quanto mais próximo for o deadline, maior a prioridade.

A escolha de um algoritmo de escalonamento de processos é importante?

- Sim e muito, pois o desempenho das aplicações depende fortemente do algoritmo, assim, sua escolha deve ser feita seguindo critérios rígidos. Sendo assim, sua avaliação de escolha pode ser feita por modelagem determinística, simulação e implementação.

O que é uma modelagem determinística?

- É algo mais matemático onde tem-se uma formula e ao dar valor determinísticos (que conhecemos), teremos valores exatos da performance desse algoritmo, mas para isso precisamos saber das entradas e como ele funciona.

Como se avalia um algoritmo de escalonamento através de simulação?

- Adicionando alguns eventos aleatórios, permitindo simular esse sistema, como um software de simulação que vai gerar algumas filas etc.

Como funciona a implementação do algoritmo de escalonamento no S.O.?

- Apesar de ser bastante custoso, é a melhor maneira de avaliar um algoritmo de escalonamento de processos, pois de fato implementa o algoritmo no S.O.

O que é uma Thread?

- É um fluxo de execução, onde um processo possui ao menos uma thread. Processos podem possuir mais de uma thread, chamados multithread, porém cada thread possui sua própria pilha e próprio registradores. E cada thread pode executar uma tarefa especifica dentro do processo.

Qual a finalidade de usar uma thread?

- Decompor um processo em partes menores com custo menor de troca de contexto. Ou seja, não troca o processo nem o contexto, pois a thread é como se fosse um “micro processo”. Sendo assim, é possível manter uma parte do programa executando enquanto outra parte estiver bloqueada esperando uma operação de E/S.

Quais os 2 níveis aos quais as threads podem ser gerenciadas?

- Em nível de usuário, onde o S.O. não enxerga que existe threads dentro de um processo e em nível de kernel (nível de S.O.) o S.O. enxerga tanto processos tanto threads, gerenciando os dois.

Quais são os modelos de threads?

- Modelo N:1 (user-level threading), onde o processo pode ter múltiplas threads porem o S.O. só enxerga uma única, que é o processo em si, o que torna simples a gerencia de threads, porém quando há uma thread de E/S, ela vai parar todas as demais.
- Modelo 1:1 (kernel-level threading), onde cada thread no nível de usuário tem uma correspondência no nível do S.O., escalonando assim as threads como os processos. É vantajoso pois assim ele não bloqueia as demais threads em caso de E/S, porém é desvantajoso no caso de permitir ao usuário utilizar todos os recursos do S.O. criando muitas threads, além de tornar mais complexo o gerenciamento.

- Modelo N:M (user-level threading), onde o usuário pode criar múltiplas threads, porém tem um limite de número de threads que podem ser geridas pelo S.O. e é vantajoso pois cada thread continua sendo vista individualmente pela S.O. e o usuário pode criar quantas threads sentir vontade.

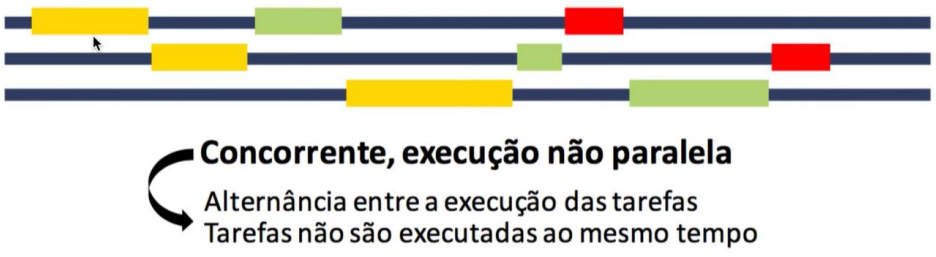
Como é feito o escalonamento de threads?

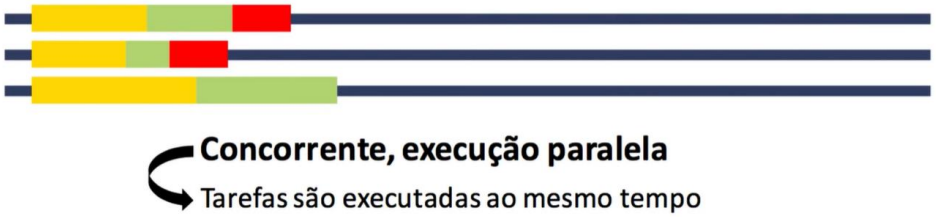
- Pode ser realizado de duas maneiras, uma no processo, em nível de processo, ou seja, o S.O. não decide quem vai escalonar, mas sim sua aplicação, e a nível de sistema, onde o processador que define qual thread vai ser executada ou não (modelo 1:1). Caso haja mais de um processador, existe a abordagem assimétrica, onde um processador mestre manipula as estruturas de dados do sistema e a abordagem simétrica onde cada processador é responsável pelo seu próprio escalonamento.

Programação concorrente

Qual a diferença entre concorrência e paralelismo?

- A concorrência é uma condição do sistema onde as tarefas (processos) estão **logicamente** ativas em um determinado momento (mas na verdade não está) e no paralelismo as tarefas são **fisicamente** executadas ao mesmo tempo. Segue um print da diferença:





Quais as vantagens e desvantagens da programação concorrente?

- Maior desempenho, melhor uso dos recursos computacionais (processador e memória) e alta capacidade de processamento (robustez). Por outro lado, não é simples programar de maneira concorrente, um dos fatores que influencia nesse quesito, é que a ordem em que processos concorrentes executam não é determinística.

O que os processos cooperativos podem compartilhar entre si?

- Um espaço de endereçamento logico e dados através de arquivos ou mensagens.

O que seria uma seção crítica na programação concorrente?

- É um segmento de código onde distintos processos acessam memoria ou arquivo compartilhados podendo causar uma condição de corrida, como por exemplo variáveis comuns, atualizar tabela compartilhada, gravar um arquivo especifico, etc. Sendo assim, apenas um processo pode executar a seção crítica de cada vez.

Quais os problemas que uma solução para o problema da seção crítica deve satisfazer?

- Somente um processo por ver acessa a região critica (mutualmente exclusivo), ser independente do numero e velocidade de CPUs, um processo fora da região crítica não bloqueia outro processo e nenhum processo espera indeterminadamente para acessar a região crítica.

Quais as soluções existentes para o problema da seção crítica?

- Soluções algorítmicas (como a solução de Peterson), soluções de hardware (Os processos podem comunicar-se uns com os outros, utilizando primitivas de comunicação entre processos, como semáforos, monitores e mensagens.) e objetos do sistema operacional.

O que é o Mutex?

- Mutex é um tipo abstrato de dados composto de um valor lógico e uma fila de threads. Ele é um tipo de sincronização de processos, sendo uma abstração muito usada para proteger regiões criticas que evita a espera ocupada. Uma variável do tipo mutex pode assumir 2 valores, livre ou ocupado. E duas de suas operações são o lock(m) que solicita acesso a região crítica e o unlock(m) que libera a região crítica.

O que é um semáforo?

- É um tipo abstrato de dado composto por um valor inteiro e uma fila de processos, que diferente do mutex, permite até que n elementos entre na região crítica por vez. Tera 2 funções, a wait(S), que espera até que S seja maior que 0 então subtrai 1 de S e signal(S) que incrementa S em 1 unidade.

O que são variáveis condicionais?

- São variáveis que não armazenam valores específicos, mas representam condições que podem ser guardadas por alguns processos. E assim, deixa todos passarem ao mesmo tempo.

O que são monitores?

- Uma forma mais alto nível para sincronização, onde permite escolher a forma que deseja escolher, como semáforo, mutex etc. Monitores simplesmente monitoram e deixa que o S.O. decida o que vai ser realizado.

Tratamento de deadlocks

Qual a sequência que o processo obedece ao solicitar recursos?

- Ele primeiro solicita (e caso a solicitação não possa ser atendida imediatamente, ele espera até poder adquirir o recurso), logo então ele usa e por final ele libera o recurso.

O que seria uma espera circular?

- Seria uma trava, onde um primeiro processo aguarda um segundo e esse aguarda um terceiro e por fim o terceiro aguarda o primeiro, ou seja, um deadlock, uma trava onde não é possível sair desse sistema de bloqueio sozinho.

Quando uma situação de deadlock pode ocorrer?

- Quando as quatro condições a seguir ocorrerem simultaneamente: Exclusão Mútua (dois processos não podem usar o mesmo recurso ao mesmo tempo), Posse e Espera (aguardar o outro processo liberar o recurso), Não-Preempção (os recursos concedidos não podem ser removidos de maneira forçada) e Espera Circular (onde um processo aguarda outro e esse outro o primeiro). Caso um recurso possua uma única instância, significa que pode ter um deadlock.

Como funciona um grafo de alocação de recursos?

- Uma aresta (i,j) direcionada do processo Pi para o recurso Rj é indicada por: Pi -> Rj (Aresta de requisição - O processo requisitou o recurso);
- Uma aresta (i,j) direcionada do recurso Rj para o processo Pi é indicada por: – Rj -> Pi (Aresta de atribuição - Uma instância do recurso está alocado ao processo).

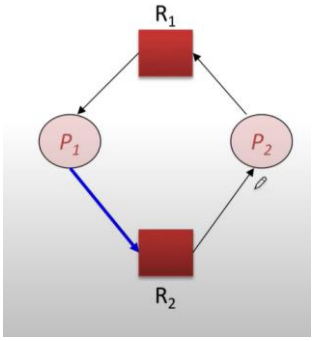
Quais as três maneiras de lidar com um deadlock?

- Prevenindo (usando protocolo que garante que um sistema nunca entrará no estado), remediando (permitindo que o sistema entre no estado de deadlock, mas consegue detectá-lo e repará-lo) e ignorando (assumindo que deadlocks nunca ocorrerão no sistema).

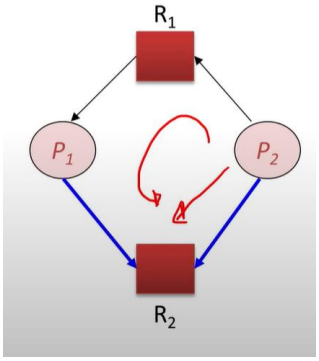
Diferencie estados seguros de estados inseguros.

- Estado seguro é aquele onde achamos uma sequencia onde todos processos são cumpridos com a quantidade disponível de instancias de um recurso, e o inseguro é quando não há essa sequência.

Exemplo de grafo de alocação de recursos onde, respectivamente, há e não há uma possibilidade de deadlock:



Não há ciclo



Há ciclo

O que é o algoritmo do banqueiro?

- É um algoritmo que é útil em sistemas onde cada recurso possui múltiplas instâncias, assim, quando um novo processo entra no sistema ele declara o número máximo de instâncias que pode precisar de cada recurso e esse número não pode ultrapassar o número total de recursos do sistema.

Comunicação entre processos

Qual a diferença entre processos independentes e cooperativos?

- Os independentes não compartilham dados com os demais processos enquanto os cooperativos compartilham algum tipo de dado com um ou mais processos (vantajoso por sua modularidade, onde pode ser dividido em várias threads e por sua conveniência, onde o usuário pode usufruir do compartilhamento).

Quais os modelos de interação?

- Um-para-um: Ligação telefônica;
-Muitos-para-um: Serviço postal, coleta de lixo;
- Um-para-muitos: Televisão, rádio, painéis publicitários;
- Muitos-para-muitos: Grupos de WhatsApp, Diário Oficial da União, Blockchain Bitcoin.

Por quem a comunicação no modelo computacional é realizada?

- Por mecanismos disponibilizados pelo S.O. ou nos ambientes de programação.

Quais as operações realizadas em um canal de comunicação?

- Criar, associar, enviar, receber, terminar e eliminar.

Qual a diferença entre canais de comunicação opacos e estruturados (estrutura interna)?

- Nos opacos, os dados têm de ser explicitamente codificados e interpretados pelos processos interlocutores, já nos estruturados, a comunicação impõe uma estrutura fixa.

Qual a diferença das orientações Mensagens-pacote e Streams dos canais de comunicação (estrutura externa)?

- Na mensagens-pacote a comunicação é realizada através de troca de mensagens (tem delimitação) e nas Streams a comunicação processa-se através da escrita e da leitura de sequencias ordenadas de bytes (não tem delimitação).

Quais as três classes de canais de comunicação?

- Memória compartilhada, caixas de mensagens e conexões virtuais (stream).

Como funciona a memória compartilhada?

- Um grupo de processos pode manipular a mesma área de memória e as áreas de memória compartilhada podem ser mapeadas no contexto de cada processo naturalmente, em diferentes espaços virtuais. Porém como não é possível antecipar em qual endereço virtual o processo será alocado, o uso de memória compartilhada não permite trabalhar com estruturas do tipo listas encadeadas, por exemplo. É importante ficar atento, pois o programador é responsável por sincronizar os processos que usam memória compartilhada.

O que são PIPEs?

- É um dos métodos mais simples de conexão de processos, e é controlado pelo S.O., porém pela sua simplicidade, ele é unidirecional e Half-duplex (um escreve por vez) e é ideal para mecanismo mestre-escravo. Não é possível escrever e ler ao mesmo tempo.

Qual a diferença entre memória compartilhada e PIPEs?

- A memoria compartilhada é de caráter muitos-pra-muitos e quem é responsável pela sincronização é o programador, já os PIPEs são de caráter um-pra-um e quem é responsável pela sincronização é o S.O.

O que são Sockets?

- São os pontos finais de uma comunicação cliente-servidor. Sendo o meio que permite criar conexão entre um ou mais pontos, permitindo assim a adoção de um modelo de comunicação baseado em conexões virtuais ou em caixas de mensagem. E eles são bidirecionais, podendo então ler e escrever ao mesmo tempo e também são orientados a comunicação local e remota.

Quais os tipos de Socket?

- Os mais significativos são o Stream (TCP) e o Datagram (UDP), e os outros tipos são o sequenced packet, reliable datagram e raw.