Sistemas Operacionais

Definição e Conceitos dos Sistemas Operacionais

O sistema operacional é parte fundamental de qualquer sistema computacional e podemos defini-lo como um software responsável por controlar o computador, cujos objetivos são gerenciar os recursos de hardware e fazer a interação entre o hardware e o software, além de ser responsável por conectar o hardware e o usuário do computador.

Breve Histórico dos Sistemas Operacionais

A evolução dos sistemas operacionais caminhou em conjunto com a evolução da arquitetura dos computadores. Vamos identificar essa evolução e dividi-la em gerações. Explore a linha do tempo.

- **1945 a 1955:** A primeira geração de computadores apresentava máquinas compostas por válvulas e painéis de programação através dos quais eram realizadas as programações. Não existiam os sistemas operacionais e nem linguagens de programação e as máquinas basicamente realizavam cálculos matemáticos como logaritmos, sendo usadas para fins militares.
- **1955 a 1965:** A segunda geração de computadores apresentava máquinas de grande porte, os mainframes, com transistores e sistema em lote. Os mainframes, possuíam sistemas operacionais e operavam através de Jobs (programa ou conjunto de programas). Os programas eram escritos em papel e depois passados para os cartões perfurados.
- **1965 a 1980:** A terceira geração de computadores apresentavam circuitos integrados e multiprogramação, cujo objetivo era permitir vários programas executarem ao mesmo tempo compartilhando os recursos de memória. Nesta época foi desenvolvido um sistema operacional que suportasse vários usuários conectados ao mesmo tempo, conhecido como Multics.
- **A parte de 1980:** A quarta geração apresenta os computadores pessoais com o desenvolvimento dos circuitos integrados ou microchip (circuito eletrônico). Os sistemas operacionais usados nesta geração foram o MS-DOS e o Unix. Iniciou-se a interface gráfica e o MS-DOS foi a base da evolução para o conhecido Windows. Surgiram os sistemas operacionais de rede e os multiprocessadores.

Principais Funções dos Sistemas Operacionais

Segundo Tanenbaum (2003), as funções dos sistemas operacionais são estender a máquina e gerenciar os recursos.

- **Estender a Máquina (ou Máquina Virtual):** A função do sistema operacional como uma máquina estendida ou máquina virtual é de esconder do programador a complexidade do hardware, apresentando uma interface amigável e gerenciável do sistema.
- **Gerenciar os Recursos:** O sistema operacional controla de forma ordenada e compartilhada os recursos do computador.

Principais Serviços dos Sistemas Operacionais

A seguir, conheça os serviços oferecidos pelo sistema operacional ao próprio sistema e aos usuários.

1. Fornece meios para que um programa seja carregado em memória e executado.

2. Possui um sistema de arquivos, permitindo criar, ler, escrever e excluir arquivos.
3. Oferece uma interface de acesso aos periféricos.
4. Promove mecanismos de monitoração de recursos.
5. Fornece meios para armazenar/manter o estado do sistema.

Podemos observar que os sistemas operacionais evoluíram juntamente com progresso dos computadores.

Atualmente, esses sistemas executam diversas funções e oferecem serviços que auxiliam o usuário e os programadores na execução dos seu trabalho, tornando para si as funções de controlar o hardware disponível.

Evolução dos Sistemas Operacionais

Rotinas do Sistema Operacional

O sistema operacional é composto por um conjunto de rotinas chamado kernel ou núcleo do sistema operacional, que gerencia os recursos do computador. Por exemplo, ao ligarmos o computador, o kernel executa programas que inicializam o computador e um conjunto de rotinas são executadas ao mesmo tempo sem uma ordem pré-definida.

Os sistemas operacionais limitam as ações executadas pelos programas em função da segurança e estabilidade, como o acesso a memória do computador. Assim, os modos de acesso aos serviços do núcleo do computador referem-se aos privilégios de execução de um programa garantindo que a memória não seja invadida por outro programa, por exemplo. Os acessos são realizados de duas formas:

- **Modo Usuário:** Os programas podem executar ações sem privilégios, como leitura de um arquivo.
- **Modo Kernel:** O acesso pode ser realizado em modo usuário ou em modo com privilégio total no computador, como acesso ao disco.

Estruturas dos Sistemas Operacionais

Quando falamos em estruturas dos sistemas operacionais, estamos tratando da maneira pela qual o código do sistema é organizado e o inter-relacionamento entre seus diversos componentes pode variar conforme a concepção do projeto.

- **Sistemas Monolíticos:** O sistema operacional é escrito como um conjunto de módulos que são compilados separadamente e depois agrupados em um arquivo executável onde cada procedimento pode ser chamado por outro quando precisar.
- **Sistemas em Camadas:** Organiza o sistema operacional como uma hierarquia de camadas. Cada camada oferece um conjunto de funções que podem ser utilizadas pelas camadas superiores.
- **Máquinas Virtuais:** Uma máquina virtual (VM) é similar a uma máquina real. Um nível intermediário é criado entre o sistema operacional e o hardware. Esse nível cria várias máquinas virtuais independentes, onde cada qual possui uma cópia virtual do hardware.
- **Modelo Cliente-Servidor:** Nesse modelo a ideia é implementar a maior parte das funções em modo usuário. Tudo que o kernel (núcleo) faz é cuidar da comunicação entre cliente e servidor.

Classificação dos Sistemas Operacionais

A seguir, vamos classificar os sistemas operacionais.

- **Sistemas Monoprogramáveis/Monotarefa:** Esses tipos de sistemas operacionais referem-se aos primeiros computadores pessoais. Executam um único programa por vez e todos os recursos da máquina são alocados exclusivamente para o programa em execução. Isso torna a implantação desse tipo de sistema simples, pois não há a necessidade de se preocupar com a concorrência de recurso.
- **Sistemas Multiprogramáveis/Multitarefa:** Esses sistemas dividem os recursos do computador com os demais programas, com isso, a CPU não fica ociosa. Podem ser classificados pela maneira em que as aplicações são gerenciadas e pelo número de usuários que utilizam o sistema, ou seja, monousuários ou multiusuários. Em relação a maneira em que as aplicações são gerenciadas, podem ser classificadas como: Sistemas batch, de tempo compartilhado ou de tempo real.
- **Sistema com Múltiplos Processadores:** Esses sistemas possuem duas ou mais CPUs interligadas e permitem que vários programas sejam executados ao mesmo tempo. Possibilitam que um programa seja subdividido e executado ao mesmo tempo em mais de um processador. Os sistemas com múltiplos processadores podem ser classificados em sistemas fortemente acoplados e sistemas fracamente acoplados.

Quando entendemos como funcionam as estruturas internas dos sistemas operacionais, observamos uma constante evolução desses sistemas que busca atender as mais diferentes demandas, gerenciando diversos programas e aplicativos do usuário e do sistema sem comprometer a performance do computador.

Características e Composição dos Sistemas Operacionais

Com a evolução dos computadores, inserindo em sua arquitetura mais eficiência e praticidade, os sistemas operacionais estão vindo mais poderosos e com funções cada vez mais avançadas. Nesta webaula vamos conhecer as principais características e propriedades dos sistemas operacionais Unix, Linux, Windows.

Sistema Operacional Unix

O Unix foi desenvolvido em Assembly, e reescrito em linguagem C para ser adaptado a outras plataformas. É um sistema multiprogramável, multiusuário, suporta vários processadores e implementa memória virtual.

A seguir, conheça algumas características do Unix.

- Sua linguagem facilita a compreensão e a portabilidade para outras plataformas.
- Pode ser usado em diversas aplicações devido a sua flexibilidade.
- Suporte a protocolos de rede.
- Sistema de arquivos com estrutura simples.
- Interface simples e uniforme com os dispositivos de entrada/saída.

O Unix se baseia em uma estrutura monolítica, ou seja, as funções são executadas em modo núcleo. Veja como é composta essa estrutura:

- **Kernel:** Núcleo do sistema operacional que pode ser dividido em duas partes, a parte dependente do hardware e a parte independente.
- **Shell:** É o responsável pela conexão dos usuários com os sistemas operacionais através da linha de comando.
- **Sistemas de Arquivos:** Responsável pela organização dos dados que são armazenados no Unix.
- **Aplicações:** São as aplicações do usuário como editores de texto, browser de navegação web e compiladores.

Linux

Sistema operacional de código-fonte aberto desenvolvido com base das características do Minix. O Linux em si é somente o kernel do sistema operacional, para que ele funcione são necessários, por exemplo, compiladores do código-fonte.

Os programas que interagem com o kernel foram desenvolvidos pela fundação GNU. Por isso, o correto é chamar GNU/Linux.

A estrutura do GNU/Linux é baseada no modelo monolítico, possuindo as mesmas características e composição do Unix: Kernel, shell, sistemas de arquivos e aplicações. As versões do Linux são conhecidas como distribuição (kernel mais programas). Veja algumas distribuições do GNU/Linux:

- **Debian:** É uma das distribuições mais antigas, estáveis e populares servindo de base para a criação das distribuições do Ubuntu, por exemplo.
- **Ubuntu:** É uma das distribuições mais populares por ser amigável, fácil de usar, instalar e obter ajuda na resolução de problemas.
- **Mint:** É uma das distribuições preferidas dos iniciantes em Linux por ser fácil de usar.
- **Fedora:** É uma distribuição para quem deseja ter ferramentas de configurações gráficas, um desktop robusto e um servidor estável.

Windows

Segundo Machado e Maia (2007), o primeiro sistema operacional lançado pela Microsoft, em 1981, foi o MS-DOS desenvolvido com a interface em linha de comando e tinha as características de ser monoprogramável e monousuário. Conheça a evolução do sistema:

- A Microsoft decidiu dar ao MS-DOS uma interface gráfica com o usuário chamada Windows. O Windows 3.0 até a versão 3.11 não eram sistemas operacionais, mas sim, interfaces gráficas com o usuário executando sobre o MS-DOS.
- O Windows 95 foi lançado e quase todas as características da parte MS-DOS foram transferidas para a parte Windows, porém o MS-DOS ainda não havia sido totalmente eliminado. Logo após, foi lançada a versão do Windows 98 que tinha poucas diferenças em relação ao Windows 95.
- Em função das limitações e deficiências do MS-DOS, a Microsoft idealizou o Windows NT (New Technology). O objetivo do Windows NT era desenvolver um sistema operacional multitarefa para executar em ambientes monousuário e multiusuário (Tanenbaum, 2003).
- O Windows 2000 é uma evolução do Windows NT, essa versão oferta serviços orientados a ambientes distribuídos e de rede, foi escrito em linguagem C e sua estrutura é baseada em micronúcleo.
- O Windows XP se tornou uma das melhores versões do sistema operacional da Microsoft devido a mudança no visual e a estabilidade do sistema.

- O Windows Vista apresentou muitos problemas, como a transparência e recursos visuais chamativos, deixando o hardware mais lento.
- O Windows 7 foi o sistema operacional mais utilizado no mercado, sendo rápido, estável e fácil de utilizar.
- O Windows 8 foi um fracasso em função da mudança no visual devido a chegada dos dispositivos sensível ao toque.
- O Windows 10 é a versão mais recente e tem algumas vantagens como leveza, adaptaçã ode tela sensível ao toque, uma plataforma unificada aproximando os aplicativos para as plataformas existentes, dentre outras.

Componentes do Sistema Operacional

O sistema operacional tem quatro componentes:

- **Gerenciamento de Dispositivos de Entrada e Saída:** É responsável por gerenciar os comandos e interrupções geradas pelos dispositivos, tratamento de erros e fornecimento de uma interface simples e fácil de usar entre os dispositivos e o restante do sistema.
- **Gerenciamento de Processos:** É responsável por criar, finalizar, escalonar, sincronizar processos e threads. Um processo é um programa em execução.
- **Gerenciamento de Arquivos:** É responsável por gerenciar os arquivos e diretórios (criar, excluir, por exemplo).
- **Gerenciamento de Memória:** Gerencia quais partes da memória estão em uso e quais partes estão disponíveis, além de administrar a troca de processos entre memória principal e memória secundária.

Uma das maiores discussões na área de tecnologia está relacionada a: Qual o melhor sistema operacional. Podemos dizer que cada sistema possuem características próprias apresentados vantagens e desvantagens como podemos observar nessa aula.

Processos

Um dos conceitos principais em sistemas operacionais gira em torno de processos. Um processo pode ser definido como um programa em execução, porém o seu conceito vai além dessa definição.

Nos computadores atuais, o processador funciona como uma “linha de produção”, executando vários programas ao mesmo tempo de forma sequencial.

A diferença entre processos e programas é importante para que seja entendido o modelo de processo. Podemos considerar que um processo é uma atividade que contém:

- Um programa.
- Uma saída.
- Uma entrada.
- Um estado.

Para melhor entender a diferença entre processo e programa, vamos fazer uma analogia com o método de preparo de um bolo:

- A receita do bolo pode ser considerada como o programa.
- O processo é a atividade que a pessoa (processador) executa durante a preparação do bolo.
- Os ingredientes são os dados de entrada.

Criação de Processos

Os sistemas operacionais devem oferecer formas para que processos sejam criados. Existem quatro eventos que fazem com que um processo seja criado:

- **Início do Sistema:** Quando o sistema operacional é inicializado, são criados vários processos. Existem os processos de primeiro plano, que interagem com os usuários e suas aplicações, e os processos de segundo plano, que possuem uma função específica, como um processo para atualizar e-mails quando alguma mensagem chegar a caixa de entrada.
- **Execução de Uma Chamada ao Sistema de Criação por Um Processo em Execução:** Quando, por exemplo, um processo está fazendo download, ele aciona um outro processo para ajudá-lo. Enquanto um processo faz o download o outro processo está armazenando os dados em disco.
- **Uma Requisição do Usuário para Criar um Novo Processo:** Quando o usuário digita um comando ou solicita a abertura de um ícone para a abertura de um aplicativo.
- **Início de um Job em Lote:** Esses processos são criados em computadores de grande porte, os mainframes.

Término de Processos

Após a criação, os processos podem ser finalizados nas condições expostas a seguir:

- **Saída Normal (Voluntária):** Ocorre quando o processo para de executar por ter acabado seu trabalho.
- **Saída por Erro (Voluntária):** Ocorre quando o processo tenta acessar um arquivo que não existe e é emitida uma chamada de saída do sistema.
- **Erro Fatal (Involuntária):** Quando ocorre um erro de programa.
- **Cancelamento por Um Outro Processo:** Ocorre quando um processo que possui permissão emite uma chamada ao sistema para cancelar outro processo.

Hierarquia de Processos

Uma hierarquia de processos se dá quando, em alguns sistemas, um processo cria outro processo e estes ficam associados.

Estados do Processo

Os processos, ao longo do processamento, podem passar por diferentes estados. Um processo ativo pode estar em três estados:

- **Em Execução:** Um processo está em execução quando está sendo processado pela CPU.
- **Pronto:** Está pronto quando possui todas as condições necessárias para ser executado e está aguardando.
- **Espera ou Bloqueado:** Quando aguarda por um evento externo ou por um recurso para processar.

Threads

Thread é um fluxo de controle (execução) dentro de um processo. Um processo pode conter um ou vários threads que compartilham os recursos do processo. Veja a seguir algumas razões para utilização dos threads:

- Aplicações compostas por threads podem ser executadas em paralelo.
- Em relação ao desempenho, o uso de threads acelera a execução da aplicação.

Implementação de Processos

Para implementar o modelo de processos, o sistema operacional mantém uma tabela que contém informações sobre: O estado do processo, seu contador de programa, o ponteiro da pilha, a alocação de memória, o status dos arquivos abertos, entre outros que permitem que o processo reinicie do ponto em que parou (Tanenbaum, 2003).

Implementação de Threads

A implementação de threads pode ocorrer no espaço do usuário, no núcleo do sistema operacional ou em ambos (implementação híbrida):

- **Thread de Usuário:** É implementada pela aplicação do usuário e o sistema operacional não sabe de sua existência.
- **Thread do Núcleo:** É implementada e gerenciada pelo núcleo do sistema operacional.
- **Thread Híbridas:** São implementadas tanto no espaço do usuário, como no núcleo do sistema operacional.

Condições de Disputa ou Condições de Corrida

Condições de disputa ou condições de corrida ocorrem quando dois ou mais processos estão lendo ou escrevendo algum dado compartilhado e o resultado final depende das informações de quem e quando executa, podendo gerar dados inconsistentes.

Regiões Críticas

Para evitar as condições de disputa, é necessário definir maneiras que impeçam que mais de um processo leia e escreva ao mesmo tempo na memória compartilhada.

Segundo Tanenbaum (2007), para termos uma boa solução, é necessário satisfazer quatro soluções:

- Nunca dois ou mais processo podem estar simultaneamente em suas regiões críticas.
- Nada pode ser afirmado sobre o número e a velocidade de CPUs.
- Nenhum processo executando em uma região crítica pode bloquear outros processos.
- Nenhum processo pode esperar eternamente para entrar em sua região crítica.

Esses métodos são chamados de exclusão mútua. Para realizá-los, podemos lançar mão das seguintes estratégias:

- Exclusão mútua com espera ociosa.
- Dormir e acordar.
- Semáforos.
- Monitores.
- Troca de mensagens.

Exclusão Mútua com Espera Ociosa

Existem alguns métodos que determinam que quando um processo está em sua região crítica, nenhum outro pode invadi-la:

- **Desabilitando Interrupções:** Nesta solução, cada processo desabilita todas as interrupções assim que entra em sua região crítica e as reabilita antes de sair dela. Desta forma, a CPU não será alternada para outro processo.
- **Variáveis de Impedimento:** Nesta solução, o processo testa e verifica o valor dessa variável antes de entrar na região crítica, e caso o valor seja 0, o processo altera o valor para 1 e entra na região crítica. Caso o valor da variável seja 1, o processo deve aguardar até que o valor seja alterado para 0. Variáveis de impedimento não resolvem o problema de exclusão mútua e ainda mantém a condição de disputa.
- **Alternância Obrigatória:** Essa solução utiliza uma variável compartilhada, turn que indica qual processo poderá entrar na região crítica (ordem). Esta variável deve ser modificada para o próximo processo antes de sair da região crítica.
- **Solução de Peterson:** Essa solução é implementada por meio de um algoritmo que consiste em dois procedimentos escritos em C, baseado na definição de duas primitivas (enter_region e leave_region) utilizadas pelos processos que desejam utilizar sua região crítica.
- **Instrução TSL:** Segundo Tanenbaum (2003), a instrução TSL (teste and set lock, ou seja, teste e atualize a variável de impedimento) conta com a ajuda do hardware. A instrução TSL RX, LOCK copia o valor do registrador RX para LOCK. Um processo pode entrar em sua região crítica apenas no caso de LOCK ser 0. A leitura do valor de LOCK e sua restauração para 0 podem ser feitas por instruções ordinárias.

Dormir e Acordar

Para resolver uma espera ociosa, são realizadas chamadas aos sistemas sleep (dormir) e wakeup (acordar), que bloqueiam/desbloqueiam o processo em vez de gastar tempo de CPU com a espera ociosa.

A chamada sleep faz com que o processo que a chamou durma até que outro processo o desperte e a chamada wakeup acorda um processo.

Semáforos

Um semáforo é uma variável inteira que realiza duas operações. Veja abaixo:

- **Down:** Decrementa uma unidade do valor do semáforo.
- **Up:** Incrementa uma unidade ao valor do semáforo.

As instruções DOWN e UP são indivisíveis e executadas no processador. Os semáforos podem ser classificados como:

- Binários, também conhecidos como mutexes, podem receber os valores 0 ou 1.
- Contadores podem receber qualquer valor inteiro positivo, além do 0.

Monitores

Um monitor é uma coleção de procedimentos, variáveis e estrutura de dados agrupados em um módulo ou pacote.

Quando um processo chama um procedimento de um monitor, é verificado se outro processo está ativo. Caso esteja, o processo que o chamou é suspenso até que o outro deixe o monitor, caso contrário, o processo que chamou poderá entrar.

Com o uso de monitores, a exclusão mútua é garantida, pois somente um processo pode estar ativo dentro do monitor em um mesmo instante e os demais processos ficam bloqeuados até que possam estar ativos no monitor.

Troca de Mensagens

Esse método utiliza duas chamadas ao sistema:

- **Send:** Envia uma mensagem para um determinado destino.
- **Receive:** Recebe uma mensagem de uma determinada origem.

Vimos nesta seção que para resolver o problema de exclusão mútua existem as seguintes soluções: Exclusão mútua com espera ociosa, dormir e acordar, semáforos, monitores, troca de mensagens. Esses métodos são equivalente quando implementados em um único processador.

Escalonamento de Processos

Em um sistema operacional, vários processos compartilham recursos ao mesmo tempo, e quem faz a escolha de qual processo deve ser executado é o escalonador de processos. Vamos, então, conhecer como é realizado o escalonamento entre processos e threads.

Escalonador de Processos

O escalonador de processos é responsável pela escolha de qual processo executar. Essa escolha é feita por meio da aplicação de algoritmos ou políticas de escalonamento, para otimizar a utilização do processador, definindo o processo que ocupará a CPU. Além de escolher o processo a ser executado, o escalonador deve prezar pelos critérios e pelos objetivos.

Diferentes sistemas operacionais apresentam características de escalonamento distintas. Por exemplo: Os sistemas operacionais em tempo real priorizam as aplicações críticas, os sistemas operacionais de tempo compartilhado alocam todos os processos com tempo igual para acesso a CPU.

A seguir, conheça as principais situações que levam ao escalonamento, segundo Tanenbaum (2003):

- **A Criação de um Novo Processo:** É necessário a escolher entre executar o processo pai ou o processo filho.
- **O Término de um Processo:** Quando um processo é finalizado, é necessário escolher outro a ser executado.
- **Bloqueio de Processo:** Quando um processo é bloqueado e está aguardando uma entrada/saída, é necessário escolher outro processo.
- **Interrupção de Entrada/Saída:** Se a interrupção for gerada por um dispositivo que finalizou a execução, o processo passará de “bloqueado” para “pronto” e o escalonador

deverá escolher entre continuar executando o processo atual ou o que acabou de ficar pronto.
- **Interrupções de Relógio:** A cada interrupção do hardware de relógio pode haver um escalonamento de processos.

Vale destacar, ainda, que existem três ambientes diferentes de escalonamento:

- **Lote:** Como não existem usuários aguardando uma resposta, tanto algoritmos preemptivos como não-preempitivos são aceitáveis para sistemas em lote.
- **Botão:** Nos sistemas interativos, a preempção se faz necessária para que outros processos tenham acesso à CPU.
- **Tempo Real:** Os processos, ao utilizarem a CPU, fazem seu trabalho rapidamente e bloqueiam, dando oportunidade para outros processos executarem.

Escalonamento de Threads

Da mesma forma que processos são escalonados, threads também são. O escalonamento de threads depende se elas estão no espaço do usuário ou do núcleo:

- **Threads de Usuário:** O núcleo não sabe de sua existência e o sistema operacional escolhe um processo “A” para executar, dando a ele o controle de seu quantum. O escalonador da thread A escolhe qual thread deve executar, por meio dos algoritmos de escalonamento descritos anteriormente.
- **Threads de Núcleo:** O sistema operacional escolhe uma thread para executar até um quantum máximo e, caso o quantum seja excedido, o thread será suspenso.

Vimos nesta aula que o escalonador de processos é responsável pela escolha de qual processo executar e quais as principais situações que levam ao escalonamento, e conhecemos os ambientes em que o escalonamento acontece. Observamos ainda que o escalonamento também ocorre nas threads.

Arquivos e Sistemas de Arquivos

As aplicações armazenam e recuperam informações durante seu processamento e um processo deve ser capaz de ler e gravar grandes volumes de informações em disco rígido, pendrives, dentre outros, além de dividir as informações com outros processos.

Veja que, segundo Tanenbaum (2003), existem três requisitos essenciais para o armazenamento da informação a longo prazo:

- Deve ser possível armazenar um grande volume de informações (a exemplo de sistemas bancários, companhias aéreas, dentre outros).
- A informação deve sobreviver ao término do processo que a usa (armazenamento em banco de dados).
- Múltiplos processos têm de ser capazes de acessar a informação concorrentemente (a informação deve ser independente de qualquer processo).

O sistema operacional organiza e estrutura essas informações por meio de arquivos.

Segundo Tanenbaum (2003), um arquivo pode ser definido como um mecanismos de abstração, oferecendo meios de armazenamento de dados e permitindo que esses dados sejam lidos posteriormente.

Nomeação de Arquivos

Segundo Tanenbaum (2003), quando um processo cria um arquivo, é atribuído a ele um nome, e quando o mesmo é encerrado, o arquivo continua existindo e outros processos têm acesso a ele, buscando-o pelo nome.

As regras de nomeação de arquivos variam de acordo com o sistema operacional.

Segundo Machado e Maia (2007), a identificação do arquivo é composta por duas partes separadas com um ponto, sendo que a parte após o ponto é denominada extensão do arquivo e identifica o conteúdo do arquivo. O quadro a seguir apresenta algumas extensões de arquivos:

Extensão	Significado
arquivo.txt	Arquivo de textos
arquivo.zip	Arquivo comprimido
arquivo.jpg	Imagem codificada com o padrão JPEG
arquivo.c	Programa fonte em C
arquivo.bak	Arquivo de cópia de segurança
arquivo.pdf	Arquivo no formato portátil de documentos
arquivo.mp3	Música codificada de áudio MPEG

Estrutura dos Arquivos

Segundo Tanenbaum (2003), os arquivos podem ser estruturados de várias formas, sendo que as três mais comuns são: Sequência estruturada de bytes, sequência de registro de comprimento fixo e árvore de registros.

Veja a seguir para saber mais sobre essas três formas:

- **Sequência Estruturada de Bytes:** Nessa estrutura, o sistema operacional não sabe o que contém o arquivo, e tudo o que ele vê são bytes. A sequência estruturada de bytes oferece flexibilidade, uma vez que os programas de usuário podem dar o nome que quiser aos seus arquivos e inserir o conteúdo que desejar.
- **Sequência de Registro de Comprimento Fixo:** Neste caso, um arquivo é uma sequência de registros de tamanho fixo, cada um com uma estrutura interna. O objetivo é que a operação de leitura retorne um registro e a operação de escrita sobreponha ou anexe um registro.
- **Árvore de Registros:** Nesta estrutura um arquivo é formado por uma árvore de registros, não necessariamente do mesmo tamanho, cada um contendo um campo-chave em uma posição fixa do registro. A árvore é ordenada pelo campo-chave para que se busque mais rapidamente por uma chave específica. Novos registros podem ser inseridos ao arquivo e o sistema operacional decide onde colocá-los.

Tipos de Arquivos

Diferentes tipos de arquivos são suportados pelos sistemas operacionais, tais como arquivos regulares, diretórios, arquivos especiais de caracteres e especiais de blocos.

Segundo Tanenbaum (2003), os arquivos regulares contêm informações do usuário e são classificados em:

- **Arquivos ASCII:** São formados por linhas de texto. A grande vantagem do arquivo ASCII é que eles podem ser mostrados e impressos como são e podem ser editados com qualquer editor de textos, além de facilitar a conexão entre a saída de um programa e a entrada de um outro.
- **Arquivos Binários:** Possuem alguma estrutura interna, conhecida pelos programas que os usam. Os sistemas operacionais devem reconhecer pelo menos um tipo de arquivo: Seu próprio arquivo executável.

Os arquivos de diretórios mantêm a estrutura do sistema de arquivos. Já os arquivos especiais de caracteres são relacionados à entrada/saída e usados para modelar dispositivos de E/S, como terminais, impressoras e rede. Os arquivos especiais de blocos são usados para modelar discos.

Acesso aos Arquivos

Os arquivos podem recuperar informações de diferentes formas, de acordo com sua organização.

Segundo Machado e Maia (2007), os primeiros sistemas operacionais armazenavam os arquivos em fitas magnéticas e seu acesso era realizado de forma sequencial, os arquivos são lidos sequencialmente, a partir do início do arquivo, sempre na ordem que os registros foram gravados.

Com a criação dos discos magnético, surgiu o acesso direto, cuja leitura/gravação é feita na sua posição por meio do número do registro. Não existem restrições em relação à ordem em que os registros são lidos ou gravados, sendo sempre necessário especificar o número do registro.

O acesso direto somente é possível quando os registros do arquivo são de tamanho fixo, sendo que o mesmo pode ser combinado com o acesso sequencial, sendo possível acessar qualquer registro de um arquivo e assim, acessar os demais.

Atributos dos Arquivos

Segundo Machado e Maia (2007), os atributos são informações de controle de cada arquivo. Os atributos variam de acordo com o sistema de arquivos, porém, o tamanho do arquivo, a proteção, a identificação do criado e a data de criação estão presentes em quase todos os sistemas.

Alguns atributos especificados na criação de arquivos não podem ser modificados, como a organização e data/hora da criação. Outros atributos podem ser alterados pelo sistema operacional, como o tamanho e a data/hora do último backup realizado. Ainda existem atributos que podem ser modificados pelo usuário, como proteção do arquivo, tamanho máximo e senha de acesso.

Sistemas de Diretórios

Segundo Machado e Maia (2007), a estrutura de diretórios é como os arquivos são organizados logicamente em disco. O diretório é uma estrutura de dados que possuem entradas associadas aos arquivos, sendo que cada entrada possui os atributos de localização do arquivo, nome, dono, organização, dentre outros.

Diretórios Simples

Uma forma simples de sistema de diretório é manter um diretório contendo todos os arquivos, chamado de diretório raiz (Tanenbaum, 2003). Esse formato era utilizado nos primeiros computadores pessoais por haver apenas um usuário, e, como vantagem, apresenta a simplicidade e a capacidade de encontrar rapidamente os arquivos.

Sistemas de Diretório Hierárquico

Com o objetivo de evitar conflitos causados por diferentes usuários escolhendo o mesmo nome, é necessário dar um diretório privado para cada um. Assim, os nomes escolhidos por um usuário não interfeririam nos nomes escolhidos pelo outro, podendo existir arquivos com o mesmo nome em dois ou mais diretórios. Esse esquema é chamado de sistema de diretórios em dois níveis.

Dessa forma, faz-se necessário agrupar esses arquivos segundo as necessidades do usuário. Assim, é necessária uma hierarquia geral, ou árvore de diretórios (folder tree) para permitir que os usuários tenham tantos diretórios quanto precisem para agrupar os seus arquivos.

Nomes de Caminhos dos Diretórios

Segundo Tanenbaum (2003), quando o sistema de arquivos é organizado por meio de uma árvore de diretórios, é necessário definir uma forma de especificar o nome dos arquivos. Para isso, são usados dois métodos: Nome de caminho absoluto e nome de caminho relativo.

Observe a seguir e saiba mais sobre esses dois métodos:

- **Nome de Caminho Absoluto:** É formado pelo caminho entre o diretório-raiz e o arquivo. Os nomes de caminhos absolutos sempre iniciam no diretório-raiz e são únicos. Como exemplo, o caminho */usuário/meus_documentos/atividades.txt* significa que dentro do diretório-raiz possui um diretório chamado “usuário”, dentro do diretório “usuário” existe um subdiretório chamado “meus_documentos” e dentro do subdiretório “meus_documentos” existe um arquivo chamado “atividades.txt”. No Windows, os componentes do caminho são separados por \. No Unix são separados por /.
- **Nome de Caminho Relativo:** É usado em conjunto com o conceito de diretório atual ou diretório de trabalho. Desta forma é possível ser designado pelo usuário um diretório como diretório atual de trabalho, em que todos os nomes de caminhos não comecem no diretório raiz. Como exemplo, se o diretório atual foi */usuário/meus_documentos*, logo o arquivo cujo caminho absoluto for */usuário/meus_documentos/atividades* pode ser referenciado apenas como *atividades*.

Sistemas de Diretórios do Linux

No Linux, o sistema de diretórios é baseado numa estrutura hierárquica e os diretórios são implementados por meio de arquivos. As partições e os discos fazem parte do diretório-raiz ou “/”. Dentro do diretório-raiz existem diretórios que possuem funções distintas.

Sistemas de Diretórios do Windows

Segundo Tanenbaum (2003), o sistema de arquivos NTFS é baseado numa estrutura hierárquica. No Windows, o sistema de arquivos está concentrada em pastas ou diretórios C:, D:, dentre outras.

Implementação e Segurança de Sistemas de Arquivos

Segundo Machado e Maia (2007), o sistema operacional precisa controlar quais as áreas ou blocos no disco estão livres quando um arquivo é criado. A seguir, serão apresentados os principais métodos de implementação de arquivos utilizados nos sistemas operacionais.

Alocação Contígua

Segundo Tanenbaum (2003), a implementação de arquivos por alocação contígua é o método mais simples, em que os arquivos são armazenados de forma sequencial no disco. Assim, se você tem um disco rígido com blocos de tamanho 1 MB, por exemplo, e um arquivo cujo tamanho seja de 40 MB, você utilizará 40 blocos sequenciais de disco para alocar o arquivo, e assim por diante.

A implementação deste método é simples e possui um bom desempenho. Porém, como desvantagem, gera a fragmentação do disco, ou seja, os arquivos ficam espalhados por todo o disco rígido do computador, o que compromete o desempenho e ainda é necessário especificar o seu tamanho do arquivo durante a criação.

Alocação por Lista Encadeada

Segundo Machado e Maia (2007), outro método de armazenamento de arquivos consiste em organizar os mesmos como um conjunto de blocos ligados logicamente no disco, independente de sua localização física. Cada bloco contém um ponteiro para o bloco seguinte do arquivo.

Apresenta a implementação de alocação por lista encadeada. Nela são apresentados quatro arquivos com seus respectivos nomes e o bloco onde começará a alocação do arquivo.

Alocação por Lista Encadeada Usando uma Tabela na Memória

Segundo Tanenbaum (2003), neste método cada palavra de ponteiro de cada bloco de disco é inserida em uma tabela na memória principal, chamada de FAT (File Allocation Table).

Apresenta a implementação de alocação por lista encadeada usando uma tabela na memória, onde são apresentados quatro arquivos com seus respectivos nomes e o bloco onde começará a alocação do arquivo.

A vantagem deste método é que o acesso aleatório se torna mais fácil pela tabela estar carregada na memória. Porém, como desvantagem, a tabela deve estar na memória o tempo todo para funcionar.

I-Nodes

Segundo Tanenbaum (2003), nesse método, cada arquivo é associado a uma estrutura chamada i-node (index-node), relacionando os atributos e os endereços em disco dos blocos de arquivos.

Com o i-node é possível encontrar todos os blocos de arquivos. Por exemplo, o arquivo A.txt foi associado a uma tabela i-node, com os seus atributos e endereços de disco, que fazem referência aos blocos de arquivos.

Nesse método, a tabela precisa ser carregada somente quando o arquivo correspondente estiver aberto. Uma desvantagem é que se existirem arquivos que precisam crescer além do esperado, seria necessário ter dois ou mais endereços de disco apontando para outros blocos de disco cheios de endereços.

Implementação de Arquivos e Diretórios

Segundo Tanenbaum (2003), para localizar a entrada de um diretório, o sistema operacional usa o nome do caminho do arquivo dado pelo usuário. A entrada de um diretório contém a informação necessária para encontrar os blocos de disco, que pode ser:

- Endereço do disco do arquivo.
- Número do primeiro bloco.
- Número do i-node.

A função principal do sistema de diretório é mapear o nome do arquivo na informação necessária para localizar os dados.

Gerenciamento de Espaço em Disco

O sistema operacional precisa gerenciar o espaço em disco dos blocos livres, garantindo um bom desempenho ao sistema.

Segundo Tanenbaum (2003), são utilizados dois métodos para monitorar os blocos livres: Lista encadeada de blocos e mapa de bits.

No método da lista encadeada de blocos, cada bloco contém a quantidade de espaços livres que puderem ter e possuem a localização dos blocos livres. Por exemplo, um disco de 16 GB precisa de uma lista livre de 16.794 blocos para conter 2^{24} números de blocos de disco.

Segundo Deitel e Choffnes (2005), um mapa de bits possui um bit para cada bloco físico do sistema de arquivos. O bit igual a 1 indica que o bloco está em uso e o bit 0 não está. Caso seja necessário alocar mais um bloco físico para o disco, é só percorrer o mapa de bits para encontrar o bit igual a 0.

Segurança e Confiabilidade do Sistema de Arquivos

Segundo Machado e Maia (2007), os arquivos são compartilhados com usuários e são utilizados para diversas finalidades. Assim, é necessário que o sistema operacional ofereça proteção desses arquivos para que usuários e processos que não tenham permissão consigam acessá-los.

Ainda segundo Machado e Maia (2007), um sistema de arquivos possui diferentes mecanismos de proteção:

- **Senha de Acesso:** Para ter acesso ao arquivo é necessário que o usuário a conheça e que o sistema conceda o acesso ao arquivo. Nesse caso, como um arquivo possui apenas uma senha, não é possível definir quais tipos de operações serão realizados.

- **Grupos de Usuários:** Essa proteção consiste em associar cada usuário a um grupo para compartilhar arquivos. Durante a criação do arquivo, o usuário define quais os usuários terão acesso a ele.
- **Lista de Controle de Acesso:** Consiste em uma lista associada a cada arquivo com as permissões de cada usuário. Quando um usuário tenta acessar um arquivo, o sistema operacional verifica a permissão do usuário para autorizar ou não a operação realizada.

Gerenciamento de Memória

Memória é o local de armazenamento de informações no computador e o seu gerenciamento em um sistema operacional é importante para garantir a eficiência das aplicações que rodam no navegador.

Nessa aula vamos apresentar como é realizado o gerenciamento de memória nos sistemas operacionais e suas características. Mostrar como se dá a monoprogramação sem troca de processos ou paginação e a multiprogramação com partições fixas.

Gerenciador de Memória

Na maioria dos computadores existe o conceito de hierarquia de memória, que pode ser representado da seguinte maneira:

- Uma pequena quantidade de memória cache.
- Uma grande quantidade de memória principal (RAM).
- Uma memória secundária com as informações armazenadas em disco.

A hierarquia de memória pode ser representada pela pirâmide. Quando mais alto as memórias estiverem na pirâmide, mais caras serão, possuirão alta velocidade de processamento e baixa capacidade de armazenamento. E quanto mais baixo, mais baratas serão, possuirão uma grande capacidade de armazenamento e baixa velocidade de processamento.

A hierarquia de memória é controlada pelo gerenciador de memória, que é responsável por gerenciar quais partes estão em uso e quais não estão alocando-a quando os processos precisarem, liberando-a após o término dos processos e controlando a troca de processo entre a memória e o disco quando a memória principal não é suficiente para manter todos os processos em execução.

O gerenciamento de memória pode ser dividido em duas classes:

- Sistemas que fazem troca de processos e paginação.
- Sistemas que não fazem troca de processos e paginação.

A troca de processos (swapping) carrega todo o programa para a memória principal, o executa por um determinado tempo e depois este retorna para o disco.

A paginação divide a memória em partições para a execução das aplicações de forma eficiente.

Monoprogramação sem Troca de Processos ou Paginação

Esse mecanismo de gerenciamento de memória é o mais simples, no qual somente um programa é executado por vez e a memória é compartilhada entre o sistema operacional e o programa.

A monoprogramação sem troca de processos ou paginação pode ocorrer em três formas:

- O sistema operacional está utilizando o espaço de endereçamento em RAM – modelo aplicado aos mainframes e minicomputadores.
- O sistema operacional está utilizando o espaço de endereçamento em ROM somente para a leitura – usado em alguns computadores de mão e em sistemas embarcados.
- Os drives de dispositivos estão em ROM e os programas do usuário e o sistema operacional está em RAM – modelo utilizado nos primeiros computadores pessoais (MS-DOS).

Multiprogramação com Partições Fixas

Esse mecanismo de gerenciamento de memória está presente na maioria dos sistemas operacionais modernos. Ele permite que vários processos executem ao mesmo tempo (multiprogramação) e quando um processo é bloqueado aguardando uma informação de entrada/saída, outro processo poderá utilizar a CPU, aumentando a sua utilização.

Neste método, a memória é dividida em n partições de tamanhos diferentes, podendo ser definida quando o sistema for iniciado. Quando um processo chega para ser executado, ele é inserido em uma fila associada à menor partição suficiente para armazená-lo.

A desvantagem deste método é que podem existir muitos processos aguardando para serem executados em algumas partições (como nas partições 1 e 4) e em outras filas não existe nenhum processo (como na partição 3).

Para solucionar o problema de espera na execução de um processo, podemos implementar uma fila única, assim um processo mais próximo do início da fila e que caiba na partição é carregado e executado.

Para finalizarmos é importante destacar que nas linguagens de programação, o gerenciamento da memória é fundamental, pois a tendência das aplicações dos usuários é consumir cada vez mais este recurso. Em muitas linguagens de programação não é necessário se preocupar com o gerenciamento, porém é importante que sejam conhecidas as restrições e capacidades do gerenciador de memória para uma programação eficaz.

Troca de Processos

A troca de processos (swapping) é realizado quando não existe memória principal suficiente para executar todos os programas ao mesmo tempo.

Na troca de processos (swapping) um programa é totalmente carregado em memória e executado por um tempo definido, enquanto os demais programas aguardam, em disco, sua vez de executar.

A troca de processos (swapping) traz totalmente cada processo para a memória, o executa por algum tempo e o retorna para o disco. A figura apresenta como se dá o funcionamento da troca de processo na memória principal.

Vamos exemplificar o conceito de troca de processo (swapping):

- Um computador possui uma memória de 512 MB e tem 4 processos para serem executados com tamanhos 481 MB, 508 MB, 380 MB e 369 MB, respectivamente.
- O gerenciador de memória seleciona um processo inteiro para ser executado em memória e os demais processos aguardam em disco a sua vez de executar.

- Se o primeiro processo (com o tamanho de 481 MB) for selecionado para ser executado, ele é carregado para a memória e executado por um tempo determinado.
- Assim que o tempo finalizar, o processo retorna para o disco e outro é selecionado para executar.
- Importante: Para a seleção do processo que será executado são utilizados algoritmos que utilizam de critérios para realizar a escolha.

O swapping permite um maior compartilhamento da memória principal e utilização dos recursos do sistema computacional.

Porém quando existe pouca memória RAM disponível, o sistema pode ficar dedicado à execução do swapping, deixando de realizar tarefas mais críticas, se tornando ineficiente.

Multiprogramação com Partições Variáveis

A alocação particionada variável, consiste em ajustar dinamicamente o tamanho das partições de memória quando os processos chegam para ser executados. Ou seja, cada processo utiliza um espaço necessário para executar, não acontecendo a fragmentação interna.

A vantagem das partições variáveis é a flexibilidade por não estar preso a um número fixo de partições, melhorando a utilização da memória, porém impactando o gerenciamento das trocas de processos e na alocação e liberação da memória.

Quando processos precisam consumir mais memória durante o processamento, é necessário alocar memória dinamicamente. Existem dois métodos de gerenciamento de memória com alocação dinâmica:

- Com mapa de bits.
- Com listas encadeadas.

Gerenciamento de Memória com Mapa de Bits

Neste método a memória é dividida em unidades de alocação, a qual é associada a um bit no mapa de bits. Se o valor do bit por 1, indica que a unidade está ocupada e se o bit for 0 ela está livre.

Gerenciamento de Memória com Listas Encadeadas

Este método consiste em manter uma lista encadeada de segmentos alocados e livres na memória.

Algoritmos de Troca de Processos

Para definir em qual área livre os processos serão executados por meio da lista encadeada são utilizados os algoritmos. Veja os algoritmos de alocação de memória:

- **First Fit (Primeiro que Couber):** É o algoritmo mais simples e que consome menos recurso do sistema. O gerenciador de memória procura ao longo da lista por um segmento livre que seja suficientemente grande para esse processo.
- **Next Fit (Próximo que Couber):** Este algoritmo é uma variação do first fit. A posição em que encontra o segmento de memória disponível é memorizada não precisando percorrer toda lista quando se quer alocar.

- **Best Fit (Melhor que Couber):** Percorre toda lista e escolhe o menor segmento de memória livre suficiente ao processo. Este algoritmo é mais lento uma vez que procura em toda a lista.
- **Worst Fit (Pior que Couber):** Sempre é escolhido o maior segmento de memória disponível de maneira que, quando dividido, o segmento disponível restante deve ser suficientemente grande para ser útil depois.
- **Quick Fit (Mais Rápido que Couber):** É um algoritmo rápido e mantém listas separadas por tamanhos de segmentos de memória mais solicitados disponível.

É importante ressaltar que a estratégia a ser usada pelo gerenciador de memória para a escolha de qual algoritmo utilizar irá depender de fatores, como o tamanho dos processos executados no ambiente e das áreas livres disponíveis.

Memória Virtual

A memória virtual é um espaço reservado no disco rígido do computador para ser utilizado quando a memória RAM não é suficiente para executar os processos.

A memória virtual permite que vários processos compartilhem a memória principal, possibilitando uma utilização eficiente do processador e reduzindo a fragmentação da memória principal. Machado e Maia (2007).

Paginação e Tabela de Páginas

A paginação é a técnica de gerência de memória em que o endereçamento virtual e o espaço de endereçamento real são divididos em blocos do mesmo tamanho, chamado páginas. Foi criada para fornecer um espaço de endereçamento linear sem a necessidade de adquirir mais memória física.

Os programas geram endereços virtuais e constituem o espaço de endereçamento virtal. Nos sistemas operacionais que trabalham com a memória virtual, o endereço virtual é enviado para a MMU (Memory Management Unit, em que um chip que está localizado na CPU).

A CPU gera os endereços virtuais e os envia a MMU. A MMU por sua vez envia os endereços físicos para a memória.

O endereço virtual divide-se em unidades conhecidas como páginas e sua referência na memória física são as molduras de página. As páginas e as molduras de páginas têm o mesmo tamanho e a movimentação entre disco e memória são sempre realizadas em unidades de página.

O mapeamento realizado pela MMU dá-se por meio da tabela de páginas. O objetivo da tabela é mapear páginas virtuais em molduras de página física.

A tabela de páginas contém o endereço virtual de cada moldura de página na memória física e o número da página é utilizado como um índice na tabela. Cada processo possui sua tabela própria e cada página possui uma entrada nela.

Cada página na tabela possui um bit presente/ausente. Se o bit for 0 (zero), indica uma interrupção por falta de página e caso o bit tenha o valor 1, a página está mapeada na memória. O número da moldura deve ser concatenado com os bits de deslocamento formando o endereço físico.

Algoritmos de Substituição de Páginas

Quando uma falta de página ocorre, o sistema operacional precisa escolher uma a ser removida da memória, a fim de liberar espaço para uma nova ser trazida.

Esse processo é feito pelos algoritmos de substituição de páginas que têm o objetivo de selecionar as páginas com as menores chances de serem referenciadas (utilizadas) no futuro.

Quanto menor for o tempo gasto com as recargas de páginas, mais eficiente será o algoritmo.

Há vários algoritmos de substituição de páginas, como podemos observar:

- Algoritmo de substituição de página ótimo.
- Algoritmo de substituição de página não recentemente utilizada (NUR).
- Algoritmo de substituição de página FIFO.
- Algoritmo de substituição de página de segunda chance (SC).
- Algoritmo de substituição de página relógio.
- Algoritmo de substituição de página menos recentemente utilizada (MRU).

Segmentação

Se um programa possui um número grande de variáveis, o espaço reservado para elas na tabela de símbolos pode se esgotar à medida que o compilador é executado e sobrará espaço nas outras tabelas.

Para resolver este problema temos que fornecer ao computador vários espaços de endereçamento independentes chamados de segmentos.

Cada segmento tem um tamanho dinâmico e independente dos outros (que varia de 0 a um valor máximo), permitindo que o segmento aumente ou diminua durante a execução. Os endereços são compostos pelo número do segmento e um deslocamento dentro do segmento.