Guilherme Vasco da Silva : 1603019

Henrique Moura Bini : 2046326

Juan Felipe da Silva Rangel :2046385

Introdução à SO

Lista - 01

Campo mourão 03/2021

1. As duas principais funções de um SO é de abstrair os recursos do hardware provendo interfaces abstratas e de gerenciar os recursos de hardware pelos aplicativos, resolvendo eventuais disputas e conflitos.
2. Todas as instruções são privilegiadas, pois todas são chamadas de sistema.
3. Em um SO de tempo real o tempo é uma característica essencial. Um sistema operacional de tempo real não necessariamente tem que ser super-rápido, um atributo essencial na verdade é ter um comportamento temporal previsível no melhor caso e no pior caso da execução do processo. O objetivo na criação de um SO de tempo real é que o seja possível minimizar as latências e espera não previstas, como por exemplo, acesso ao disco e sincronizações excessivas.

Os Sistemas Operacionais de tempo real podem ser divididos em duas classes, tempo real Críticos e não críticos. A principal diferença entre eles é que no crítico, a perda de um prazo pelo sistema pode perturbar seriamente o sistema físico sob seu controle, podendo gerar consequências graves, como por exemplo, o sistema de um freio ABS, no qual, caso não sejam respeitados os prazos é possível causar um acidente. Diferente do SO de tempo real crítico, em um não-crítico , a perda de um prazo é perceptível e degenera o serviço prestado, sem maiores consequências, como por exemplo em sistemas de reprodução de mídia, que em caso de atrasos podem ocorrer falhas na música que está sendo tocada.

1. O modo usuário é a parte do SO em que os programas de interface com o usuário se localizam , já o modo núcleo é executado quando uma processo deixa de executar do modo usuário e um chaveamento é feito, passando a executar no modo núcleo. Isso normalmente acontece quando uma chamada de sistema é feita. As flags que são definem o nível de privilégio só podem ser modificados por código executando no nível núcleo, o que impede que usuários maliciosos de contornar essa barreira de proteção criada.
2. O gerenciamento de recursos inclui a multiplexação (compartilhamento) de recursos de duas maneiras diferentes: no tempo e no espaço. Quando um recurso é multiplexado no tempo, diferentes programas ou usuários se revezam usando-o. Primeiro, um deles usa o recurso, então outro e assim por diante. Por exemplo, com apenas uma CPU e múltiplos programas querendo ser executados nela, o sistema operacional primeiro aloca a CPU para um programa, então, após ele ter sido executado por tempo suficiente, outro programa passa a fazer uso da CPU, então outro, e finalmente o primeiro de novo. Determinar como o recurso é multiplexado no tempo — quem vai em seguida e por quanto tempo — é a tarefa do sistema operacional. Outro exemplo da multiplexação no tempo é o compartilhamento da impressora. Quando múltiplas saídas de impressão estão na fila para serem impressas em uma única impressora, uma decisão tem de ser tomada sobre qual deve ser impressa em seguida. O outro tipo é a multiplexação de espaço.
3. É muito importante saber quais rotinas de bibliotecas resultam em chamadas de sistema levando em consideração a questão do desempenho. Chamadas ao sistema requerem desvio de fluxo e tratamento das chamadas que trazem consigo armazenamento e resgate de contexto, coisas que tomam tempo.
4. Interrupções de Hardware: tratam eventos assíncronos externos ao processador. Interrupções de Software (Trap): tratam eventos síncronos solicitados por aplicação. Exceções: tratam eventos síncronos internos ao processador.
5. Algumas das vantagens de um SO de código-fonte aberto é primeiramente, que a qualidade do SO é constantemente melhorada, com uma grande quantidade de pessoas que podem trabalhar nas resoluções de erros e contribuir com uma aprimoração de funções que já foram implementadas. Uma outra vantagem é que ele é de graça, o que pode contribuir principalmente para estudantes, que muitas vezes necessitam de determinados softwares para fins de aprendizado.
6. Ele pode ser estruturado de diversas maneiras sendo elas: Sistema Monolítico, sistema micronúcleo, sistema em camadas, sistemas híbridos, máquinas virtuais, sistema de contêineres, sistema uninúcleo, sistema exonúcleo.
7. O núcleo kernel pode mascarar (inibir) interrupções durante atividades que poderiam resultar em corrupção de dados se elas fossem permitidas, por exemplo quando se está manipulando listas. Para evitar a inibição de interrupções é evitar a mudança de modo usuário para modo supervisor, pois com o sistema em modo supervisor interrupções podem ser inibidas e reabilitadas.
8. O comando fopen é uma função de biblioteca pois a linguagem C não possui nenhum comando de entrada/saída, todas as operações de entrada/saída são realizadas através de funções de bibliotecas.

| Arquitetura | Benefícios | Deficiências |
| --- | --- | --- |
| Sistema Monolítico | Alto desempenho. | Baixa robustez.  Velocidade de desenvolvimento. |
| Sistema Micronúcleo | Maior Modularidade.  Manutenção fácil.  Velocidade de desenvolvimento | Número excessivo de chamadas de sistema, gerando menor desempenho. |
| Sistema em camadas | Domínio das redes de computadores | O empilhamento de várias camadas de software pode prejudicar o desempenho. |
| Sistema de Contêineres | Permite a criação de domínios baseado na necessidade | Diferentes domínios não podem interagir entre si. |
| Máquinas virtuais | Executar diferentes Sistemas Operacionais simultaneamente, utilizando o mesmo hardware. | Custo adicional de execução dos processos na máquina virtual se comparado com uma máquina real. |

1. O hardware não consegue distinguir entre uma função do SO e de uma aplicação qualquer porém quando o SO executa uma instrução ele chaveia o processador para o modo núcleo.
2. O sistema operacional é quem faz o carregamento, alocação e o escalonamento de um programa para que ele possa ser executado.
3. Não, pois quando um programa é compilado ele é compilado para um tipo de arquitetura específica, dessa forma caso seja executado em outra arquitetura o sistema operacional muito provavelmente não irá entender a estrutura do programa compilado.
4. Ao ligar o computador o sistema Básico de entrada/saída (BIOS) entre em execução esse sistema é pré-gravado em um chip de memória ROM, junto da BIOS também existem outros dois programas dentro da ROM: SETUP e o POST que realiza uma sequência de testes e verificações do hardware.

Após as verificações do BIOS, ele efetua a descompactação para a memória RAM do bootstrap, e então o contador de programa (PC) da CPU é direcionado para o endereço da primeira instrução do boot executando as instruções deste programa que é responsável por carregar o SO ou parte dele que for necessária para iniciar o computador.

Ao terminar de carregar o So, o PC é redirecionado para a primeira instrução do SO na RAM, após isso o SO está em execução.

**17.** Relacione as afirmações aos respectivos tipos de sistemas operacionais: distribuído (D), multiusuário (M), desktop (K), servidor (S), embarcado (E) ou de tempo real (T): [3]

[ T ] Deve ter um comportamento temporal previsível, com prazos de resposta claramente definidos.

[ S ] Sistema operacional usado por uma empresa para executar seu banco de dados corporativo.

[ E ] São tipicamente usados em telefones celulares e sistemas eletrônicos dedicados.

[ D ] Neste tipo de sistema, a localização física dos recursos do sistema computacional é transparente para os usuários.

[ M ] Todos os recursos do sistema têm proprietários e existem regras controlando o acesso aos mesmos pelos usuários.

[ E ] A gerência de energia é muito importante neste tipo de sistema.

[ K ] Sistema que prioriza a gerência da interface gráfica e a interação com o usuário

[ S ] Construído para gerenciar de forma eficiente grandes volumes de recursos.

[ K ] O MacOS X é um exemplo típico deste tipo de sistema.

[ E ] São sistemas operacionais compactos, construídos para executar aplicações específicas sobre plataformas com poucos recursos

**18.** Coloque na ordem correta as ações abaixo que ocorrem durante a execução da função printf("Hello world") por um processo (observe que nem todas as ações indicadas fazem parte da sequência).

[ 5 ] A rotina de tratamento da interrupção de software é ativada dentro do núcleo.

[ 8 ] A função printf finaliza sua execução e devolve o controle ao código do processo.

[ 2 ] A função de biblioteca printf recebe e processa os parâmetros de entrada (a string “Hello world”).

[ 3 ] A função de biblioteca printf prepara os registradores para solicitar a chamada de sistema write()

[ 7 ] O disco rígido gera uma interrupção indicando a conclusão da operação.

[ 10 ] O escalonador escolhe o processo mais prioritário para execução.

[ 4 ] Uma interrupção de software é acionada.

[ 1 ] O processo chama a função printf da biblioteca C.

[ 6 ] A operação de escrita no terminal é efetuada ou agendada pela rotina de tratamento da interrupção.

[ 9 ] O controle volta para a função printf em modo usuário.

**19.** Existem diferentes tipos de Sistemas Operacionais, os principais serão listado abaixo:

* Sistema Distribuído: Uma das características principais em um sistema Distribuído é que, os recursos de cada computador estão disponíveis a todos na rede, de forma transparente aos usuários, como por exemplo, ao executar um aplicação, o usuário interage com sua interface porém não sabe onde está executando ou armazenando seus arquivos.
* Sistema Multiusuário: Nesse tipo de sistema o SO deve suportar diferentes identificações do “dono” de cada recurso dentro do sistema, controlando e impondo regras de acesso para impedir o uso indevido de recursos por usuário não-autorizados.
* Sistema Servidor: Um sistema operacional servidor tem como sua principal característica a gestão eficiente de grande quantidades de recursos, decretando limites e prioridades sobre o uso dos recursos pelos usuários e suas aplicações.
* Sistema Embarcado: É possível dizer que um sistema embarcado é caracterizado por ser construído para operar sobre um hardware com poucos recursos de processamento, armazenamento e energia. Por esse motivo são normalmente usados em sistemas de automação e controle automotivos, equipamentos eletrônicos de uso doméstico e muitos outros.
* Sistema de Tempo Real: Ao contrário do que muitos pensam, um sistema de tempo real não precisa ser extremamente rápido, porém deve ter um comportamento temporal previsível no melhor e no pior caso. Neste SO o objetivo é minimizar as esperas e latências imprevisíveis, como tempo de acesso a disco e sincronizações excessivas.
* Sistema Desktop: Este tipo de Sistema Operacional tem como objetivo atender o usuário doméstico e corporativo no uso de atividades corriqueiras, como edição de texto, navegação na Internet e reprodução de mídia. Para que a execução dessas tarefas sejam mais práticas, o SO implementa uma interface gráfica, gerando uma maior interatividade do usuário com o sistema.

**20.** Em sistemas operacionais monoprogramados os dados da entrada do programa é carregado na memória e só são descarregados ao término da execução dessa forma caso seja necessário a espera de uma entrada/saída a CPU ficará ociosa até receber o resultado da entrada/saída.

Em sistemas operacionais multiprogramados caso um programa necessite esperar dados externos o sistema operacional suspende esse arquivo até ele receber estes dados externos enquanto esse programa está suspenso a CPU executa outros programas, quando o programa sai da suspensão ele retoma a execução da parte em que ele parou.

**21.** Buffering é um prática em que a CPU só irá executar um programa de entrada/saída quando ele terminar de colocar os dados no buffer dessa forma a CPU não fica ociosa enquanto o programa coloca os dados na memória. O buffering possibilita que dispositivos que operam em velocidades distintas possam utilizar uma região de memória intermediária para obter melhor desempenho, ele impede que um programa fique ocioso porque um programa de entrada/saída está transferindo dados para a memória.

**22.** A técnica de spooling dá a possibilidade de utilizar um recurso exclusivo no qual apenas um processo pode utilizar esse recurso por vez.

Isso ocorre através da criação de uma área de spooling em que os programas colocam tudo o que for necessário para utilizar esse recurso na área de spooling. Os arquivos na área de spooling ficam em um fila, essa fila é gerenciada pelo sistema operacional tem a função de transferir os arquivos da área de spooling para a utilização do recurso sem que o programa precise ficar verificando se a execução já foi finalizada, o sistema operacional notifica o programa apenas a finalização do uso do recurso e pego o próximo arquivo da área de spooling.

**23.** O caching melhora o desempenho identificando quais partes são usadas com maior frequência que outras, por exemplo, a maioria dos sistemas operacionais mantém parte de arquivos muito usados na memória principal para que não seja necessário ficar buscando no disco de modo repetitivo. Bem como o exemplo anterior, os resultados da conversão de nomes de rota longa como no endereço de disco onde o arquivo está localizado podem ser registrados em cache para impedir um acesso repetitivo na procura do mesmo dado, melhorando o desempenho do sistema.

**24.** Os drivers e interfaces plug-and-play provem um suporte a diversos tipos de dispositivos sem a necessidade do usuário instalar drivers específicos para cada novo dispositivo conectado, além de fazer a gerência de energia dos dispositivos conectado de forma dinâmica isso faz com que a extensibilidade do sistema operacional aumente tendo em vista que a maior parte do dispositivos irá funcionar logo após ser inserido no computador.

**25.** Uma maneira criada para evitar que um processo monopolize o processador é criando para cada atividade um prazo de processamento, denominada fatia de tempo ou quantum. Caso o quantum de uma atividade for esgotado, a tarefa em execução perde o processador e volta para uma fila de tarefas “prontas”, que estão na memória aguardando uma nova oportunidade de executar, e outra tarefa é selecionada para consumir o processador. Esse ato de retirar um processo da CPU “à força” é chamado de preempção.

Sistema que implementam esse conceito, utilizam a preempção por tempo, que usa as interrupções gerador por um temporizador programável disponível no hardware (temporizador de intervalo). O temporizador anteriormente citado é programado para gerar interrupções em intervalos de tempo regulares que são recebidas por um tratador de interrupções e encaminhadas ao núcleo. Quando o processador recebe uma tarefa, o núcleo ajusta um contador de ticks que essa tarefa pode usar, ou seja , seu quantum.

A cada tick esse contador é decrementado e quando ele chegar a zero, a tarefa perde o processador e volta à fila de tarefas, como citado anteriormente.

**26.** A EFI suporta um interpretador de comandos, por meio do qual usuários podem acessar diretamente dispositivos de computador, e incorpora unidades de dispositivos para suportar acesso a unidades de disco rígido e redes, imediatamente após ligar o sistema. Isso proporciona maior funcionalidade na hora da inicialização, pois uma BIOS típica contém somente instruções de baixo nível que proporcionam funcionalidade limitada e restringem o modo como o software é carregado inicialmente.

**27.** O POSIX (Portable Operating System Interface — interface portátil para sistemas operacionais) é um padrão do UNIX desenvolvido para tornar possível escrever programas que possam ser executados em qualquer sistema UNIX. Para isso, o POSIX define uma interface minimalista de chamadas de sistema à qual os sistemas UNIX em conformidade devem dar suporte, sendo que hoje em dia até outros sistemas operacionais também dão suporte à interface.

**28**. Para o carregamento de bibliotecas compartilhadas, mapeamento da memória e no final do rastreio, é necessário fazer a emissão das informações sobre a data para a saída padrão.

**29.** É possível observar que as chamadas de sistemas vistas no comando strace date aparecem novamente no comando ltrace date porém na forma de chamada de biblioteca