
Sabrina Carlos Costa

**1.** Quais são as duas maneiras de análise dos SO´s? Explique

- Máquina Estendida (Top-Down): Essa abordagem enfoca como os sistemas operacionais tornam as operações de baixo nível (como acesso a hardware) mais acessíveis ao usuário. Isso significa que o SO abstrai a complexidade das interações com o hardware, permitindo que os usuários realizem tarefas sem precisar entender todos os detalhes técnicos. O objetivo é melhorar a experiência do usuário, facilitando a utilização do sistema.
- Gerenciador de Recursos (Bottom-Up): Essa perspectiva se concentra no papel do sistema operacional em gerenciar os recursos do computador, como CPU, memória, e dispositivos de entrada/saída. O SO atua como um intermediário entre o hardware e os aplicativos, garantindo que os recursos sejam alocados de forma eficiente e que diferentes processos possam operar de maneira harmoniosa, sem interferir uns nos outros.

**2**. Conceitue:

**a)** Multi-Programação

A multiprogramação é uma técnica que permite a execução simultânea de múltiplos processos na memória, dividindo-a em várias partes e alocando a cada parte um job (ou tarefa). O principal objetivo é maximizar a utilização do processador, mantendo uma quantidade suficiente de jobs na memória para que, enquanto um processo aguarda por operações de entrada/saída (E/S), o processador possa alternar para outro processo e assim minimizar o tempo ocioso.

Aspectos importantes:

- Proteção de memória: O hardware desempenha um papel fundamental na proteção dos processos, impedindo que um job acesse a memória de outro, o que ajuda a garantir a integridade dos dados e a estabilidade do sistema.

- Separação de funções: Em muitas arquiteturas, havia uma distinção clara entre máquinas dedicadas a processamento e aquelas responsáveis por operações de E/S, o que permitia um gerenciamento mais eficiente dos recursos.
- Interação humana: Os operadores frequentemente precisavam se deslocar entre diferentes máquinas para gerenciar os jobs, o que aumentava a complexidade da operação.

Dessa forma, a multiprogramação melhora a eficiência geral do sistema, permitindo que múltiplos processos sejam gerenciados de forma eficaz, reduzindo a ociosidade do processador e otimizando o uso dos recursos disponíveis.

**b)** Spooling

Spooling é uma técnica que permite o gerenciamento eficiente de tarefas que requerem operações de entrada/saída (E/S) em sistemas computacionais. O termo significa "Simultaneous Peripheral Operation On-Line" e se refere à capacidade de sobrepor operações de E/S com a execução de processos.

Principais características do spooling:

- Leitura de jobs: O spooling possibilita que a leitura de cartões perfurados ou arquivos de jobs seja realizada diretamente do disco, em vez de depender de dispositivos de E/S que poderiam ser mais lentos.
- Gerenciamento de jobs: Assim que um job termina sua execução, o sistema operacional pode alocar automaticamente o próximo job disponível em uma partição livre da memória. Isso reduz o tempo de espera e melhora a eficiência do processamento, já que os jobs podem ser lidos do disco enquanto outros estão em execução.
- Redução do tempo de processamento: Apesar do avanço tecnológico, o tempo de processamento ainda era crítico. Com o uso do spooling, a espera para corrigir erros de programação ou realizar ajustes era minimizada. Os programadores podiam enviar um novo job para a fila enquanto outros estavam sendo processados, em vez de aguardar que cada job fosse executado completamente.
- Eficiência em lotes: Os jobs eram tratados em lotes, permitindo que vários trabalhos fossem organizados e processados de forma sequencial. Isso otimiza a utilização dos recursos do sistema.

Dessa forma, o spooling é uma técnica essencial que melhorou significativamente a eficiência dos sistemas operacionais, permitindo que múltiplas operações de E/S fossem realizadas em paralelo com a execução de processos.

**c)** TimeSharing

Time Sharing é uma técnica de gerenciamento de sistemas operacionais que permite que múltiplos usuários acessem um computador simultaneamente, proporcionando a cada um deles a sensação de ter o sistema à sua disposição.

Características principais do Time Sharing:

- Acesso simultâneo: Cada usuário tem um terminal online conectado ao sistema, permitindo a interação em tempo real com o computador. Isso facilita o uso compartilhado dos recursos de processamento.
- Primeiro sistema Time Sharing: Um dos primeiros sistemas a implementar esse conceito foi o CTSS (Compatible Time Sharing System), que funcionava em um computador 7094 modificado. Esse sistema possibilitou a divisão do tempo de CPU entre vários usuários.
- Sensação de exclusividade: Apesar de haver múltiplos usuários, cada um tem a impressão de que está utilizando o computador exclusivamente, devido à rápida alternância entre os processos. Por exemplo, se 20 usuários estão conectados, mas apenas 3 estão ativos, o processador é alocado entre esses três jobs, permitindo que cada um receba uma parte do tempo de processamento de forma eficiente.
- Diversidade de sistemas operacionais: Diferentes máquinas podiam operar com sistemas operacionais distintos, como o OS/360 para a linha System/360 e o MULTICS da General Electric. Essa diversidade, no entanto, também resultava em problemas de incompatibilidade entre os sistemas.

O Time Sharing revolucionou a computação ao permitir que vários usuários interagissem com um único computador de forma eficiente, melhorando a utilização dos recursos e a produtividade.

**3.** Quais os tipos de Sistemas Operacionais? Caracterize cada um deles.

- Sistemas monoprogramáveis ou monotarefa: executam apenas uma tarefa por vez, como era o caso do MS-DOS. Nesse tipo de sistema, o processador fica totalmente focado em uma única tarefa até que ela termine. Isso, claro, pode deixar o sistema ocioso em alguns momentos, já que ele não pode alternar entre diferentes tarefas.
- Sistemas multiprogramáveis ou multitarefa: permitem que vários processos rodem ao mesmo tempo, alternando entre eles de forma inteligente. Isso garante que o processador fique ocupado ao máximo, distribuindo melhor os recursos, como acontece com sistemas como Unix, Linux e Windows, que usamos no dia a dia.
- Sistemas operacionais em tempo real: são feitos para situações onde o tempo de resposta é crítico, como em aviões ou equipamentos médicos. Nesses casos, o sistema precisa reagir de forma precisa e rápida. Existem os de tempo real "hard", que garantem uma resposta dentro de um prazo fixo, e os "soft", que não são tão rígidos, mas ainda precisam ser rápidos. Exemplos desses sistemas são o VxWorks e o QNX.
- Sistemas operacionais distribuídos: várias máquinas trabalham juntas como se fossem um único sistema. Isso é útil em situações que exigem muito processamento ou que precisam garantir que o sistema continue funcionando mesmo que uma das máquinas tenha algum problema. Alguns exemplos são os sistemas Amoeba e Plan 9.
- Sistemas operacionais de rede: facilitam o compartilhamento de recursos entre computadores conectados. Por exemplo, eles permitem que diferentes computadores compartilhem arquivos, impressoras e outros recursos, como acontece em servidores de empresas. Exemplos incluem o Windows Server e o Linux com NFS.

- Sistemas de tempo compartilhado (Time Sharing): dão a várias pessoas a possibilidade de usar o mesmo computador ao mesmo tempo. O sistema divide o tempo de uso do processador entre os usuários, o que faz com que cada um tenha a sensação de que está usando a máquina sozinho. Isso foi uma grande inovação, vista em sistemas como Unix e MULTICS.
- Sistemas operacionais móveis: como o Android e o iOS, foram criados especialmente para smartphones e tablets. Eles são otimizados para economizar bateria e gerenciar bem os recursos limitados desses dispositivos, além de oferecer uma experiência fluida de toque e conectividade constante.
- Sistemas operacionais embarcados: estão em dispositivos que têm funções bem específicas, como carros, eletrodomésticos e dispositivos de IoT (Internet das Coisas). Esses sistemas, como o FreeRTOS e o Embedded Linux, são feitos para serem super eficientes, já que os dispositivos onde rodam geralmente têm pouca capacidade de processamento.

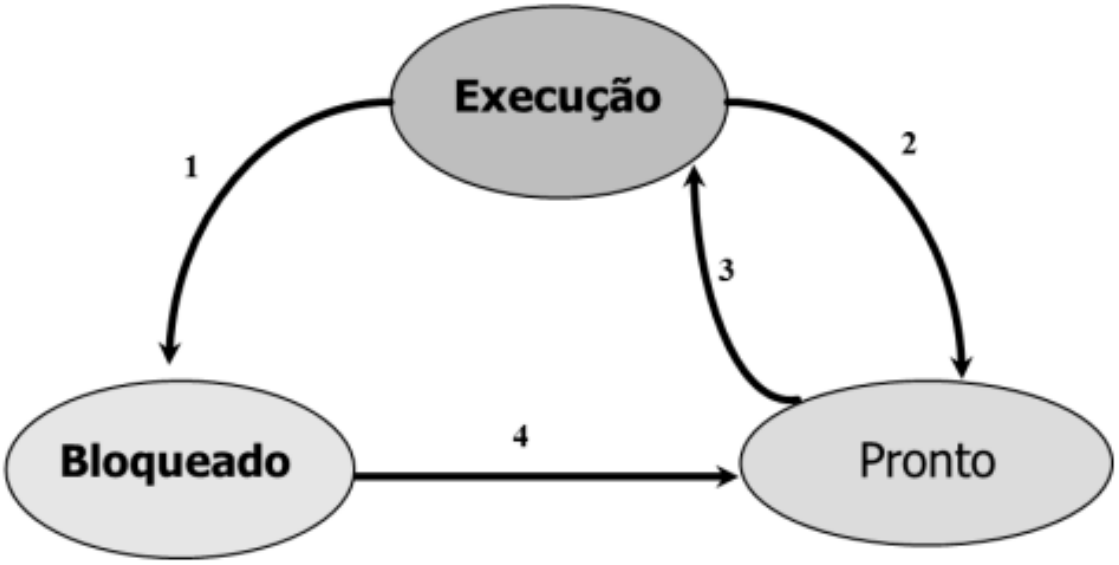
**4.** Qual a importância das System calls?
As system calls (chamadas de sistema) são a interface fundamental entre o sistema operacional e os programas de usuário. Elas consistem em um conjunto de instruções que permitem que os aplicativos solicitem serviços e recursos do sistema operacional, como gerenciamento de arquivos, manipulação de processos, e operações de entrada/saída. De modo geral, a execução de tarefas complexas de forma segura e eficiente.

**5.** Qual a importância da instrução trap?
A instrução **trap** é fundamental para sistemas operacionais, pois permite a transição controlada entre o modo de usuário e o modo de kernel. Ela possibilita que programas acessem serviços do sistema de forma segura, gerenciando recursos, tratando erros e garantindo segurança. Ao interromper a execução normal, a trap assegura que operações privilegiadas sejam realizadas pelo sistema operacional, promovendo a estabilidade e a proteção do sistema.

**6.** Defina PCB. Quais os principais dados da PCB?
PCB, ou Bloco de Controle de Processo, é uma estrutura de dados utilizada pelo sistema operacional para gerenciar informações sobre um processo em execução. Ele é fundamental para o gerenciamento de processos, pois armazena todos os dados necessários para que o sistema operacional possa controlar a execução de cada processo. O PCB é essencial para a operação do sistema operacional, pois permite o rastreamento e o gerenciamento eficaz de todos os processos em execução.
Os principais dados do PCB são:

- Ponteiros.
- Estado do processo.
- Prioridade do processo.
- Limites de memória.
- Registradores
- Estado de seus arquivos abertos.
- Lista de arquivos abertos.
- Contabilidade do processo no uso de recursos.

**7.** Explique os estados dos processos do SO. Comente cada mudança de estado.




Pronto: O processo está preparado para ser executado, mas o processador está ocupado com outro processo. Ele aguarda sua vez na fila de execução.

Execução: O processo está ativamente utilizando o processador para realizar seu processamento.

Bloqueado/Espera: O processo não pode continuar sua execução porque está aguardando a ocorrência de um evento específico, como a finalização de uma operação de E/S. Isso pode ocorrer por:

- Necessidade de entrada que ainda não está disponível.
- Realização de uma operação de E/S, que faz o processo liberar o processador e esperar pela conclusão da operação.

Um processo pode mudar de estado diversas vezes durante sua execução, em função de eventos voluntários (gerados pelo próprio processo) ou involuntários (provocados pelo sistema operacional). As principais mudanças de estado são:

1. Execução -> Bloqueado/Espera: Ocorre quando um processo em execução solicita um recurso ou espera por um evento (ex: uma operação de E/S).
2. Execução -> Pronto: Ocorre quando o tempo de execução do processo se esgota (fim da fatia de tempo) e ele precisa aguardar sua próxima oportunidade de uso da CPU.
3. Pronto -> Execução: Ocorre quando um processo na fila de prontos é alocado para o processador. Inicialmente, ao ser criado, o processo entra na fila de pronto até que consiga ser executado.
4. Bloqueado/Espera -> Pronto: Ocorre quando um evento aguardado pelo processo é concluído (ex: a E/S finaliza), permitindo que o processo retorne à fila de prontos antes de ser executado novamente.