Módulo 2 – Visão Geral de Sistemas Operacionais

João Vitor Yukio Bordin Yamashita

September 19, 2022

1 Perguntas

1. Os sistemas operacionais são importantes para gerenciar o uso de *hardware* de forma a garantir a segurança e eficiência dos processos para uso geral. (ex. Computadores); os sistemas operacionais são gerenciadores, criados para garantir certas condições específicas da aplicação, por exemplo, as prioridades de um sistema operacional de um computador são diferentes das prioridades de um sistema operacional para um microcontrolador coletando dados em um ambiente externo, as diretrizes de gerenciamento são inerentes da aplicação do sistema operacional.

2. Não, como mencionado no item anterior o SO é um gerenciador, mas podemos ter sistemas computacionais sem a necessidade desse tipo de gerenciamento, ou devido a sua simplicidade ou devido ao custo de processamento e memória gerados pelo uso de um SO. Por exemplo, um microcontrolador que lê um sensor e imprime o seu valor em um LCD pode não precisar de um SO para o gerenciamento do processo de leitura e impressão.

```
int main(){
    Sensor sensor = Sensor();
    Lcd lcd = Lcd();
    sensor.init();
    lcd.init();
    for(;;)
    {
        lcd.print(sensor.read());
        delay_ms(20);
    }
    return 0;
}
```

Listing 1: Exemplo de sistema computacional sem SO

   No exemplo acima temos um sistema computacional que funciona sem nenhum tipo de sistema operacional.

3. Um processo é um pedaço do código, com todas as suas variáveis, que está sendo executado pela CPU em um determinado período de tempo. Considerando o exemplo acima, poderíamos dizer que a leitura e impressão de um valor do sensor é um processo. No caso de um SO, podemos ter mais de um processo para a CPU executar, por exemplo, vamos dizer que temos três sensores e três LCD's, poderíamos ter um SO para realizar as leituras e impressões de cada sensor, e cada uma dessas tarefas seria um processo para a CPU executar, na Figura 1 temos uma representação desses processos, cada um teria um 'tempo' de CPU e seria controlado pelo SO.

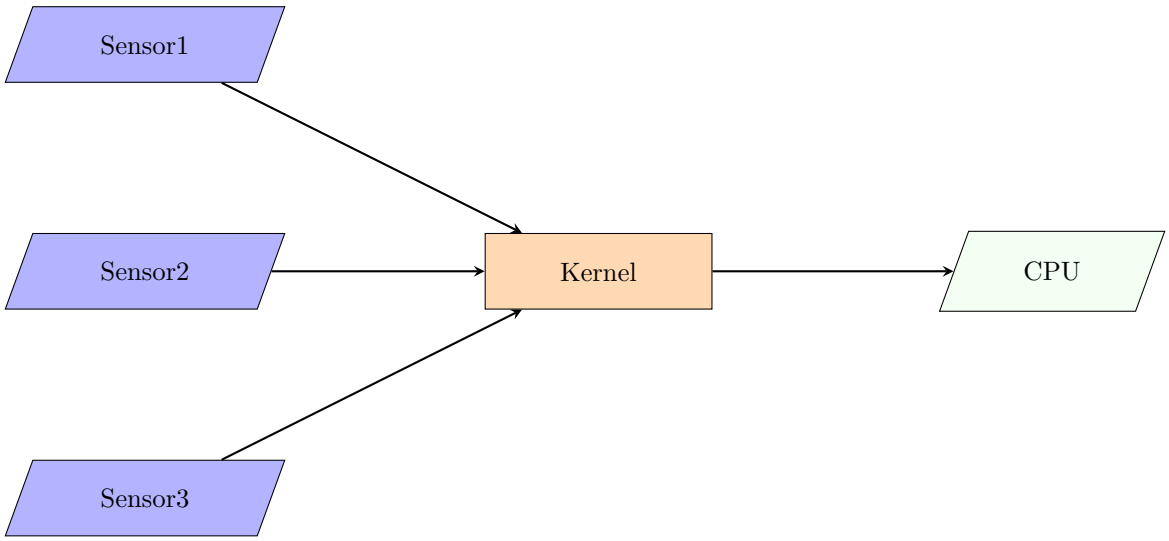

Figure 1: Gerenciamento dos processos

Os processos tem um programa, entradas, saídas e um estado, que podem ser usados e gerenciados pelo SO.

4. Gerenciar os processos, ou seja, gerenciar o uso da CPU; gerenciar memória, além de gerenciar como a memória pode ser utilizada por processos, gerenciar como uma variável ou endereço de memória pode ser utilizado, por exemplo, se dois processos usam a mesma variável, não podemos permitir que um processo ativo altere o valor dela e o outro processo quebre por causa disso, por isso o SO deve gerenciar o uso da memória e garantir que não tenhamos nenhum comportamento inesperado. Além disso o SO deve gerenciar a entrada e saída da máquina, por exemplo, agora eu estou escrevendo usando um teclado, estou usando um mouse, estou vendo a tela onde estou escrevendo e usando um fone de ouvido, o SO deve gerenciar como cada um desses periféricos serão processados e mostrados para o usuário.

5.
   - Sendo executado - o processo está usando a CPU naquele instante;
   - Pronto - o processo está pronto para ser executado;
   - Bloqueado - esperando alguma condição acontecer para voltar a ser executado.

6. A Figura 2 representa os estados mencionados no item acima:

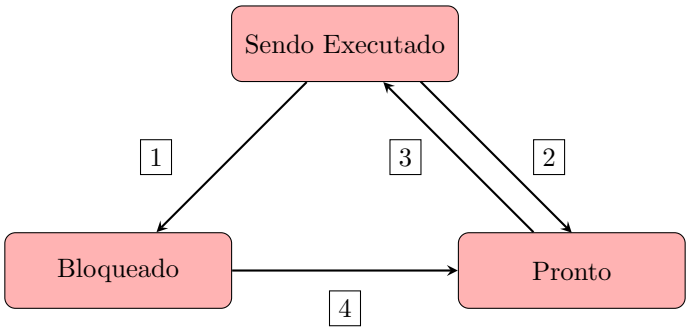

Figure 2: Estados de um processo

1 Nessa seta o processo saiu do estado executando e foi para o bloqueado, ou seja, ele está esperando alguma interrupção ou entrada para retornar a ser executado;

2 O processo foi finalizado ou haverá uma troca de contexto/processo a ser executado pela CPU e o SO escolhe outros processos devido ao tempo de execução do processo atual ter acabado;

3 Um processo é reiniciado/retomado pelo SO e será executado pela CPU;

4 A condição para reiniciar/retomar o processo bloqueado foi satisfeita e ele será executado pela CPU.

7. Os principais tipos de estruturas são:
   - Monolíticos;
   - Em camadas;
   - Máquinas Vituais;
   - Micro-kernel;
   - Clinte-Servidor

8. Significa que o escalonador de processos decidiu remover o atual processo em execução para um estado de bloqueado/pronto para executar outro processo mais crítico para o sistema; por exemplo, podemos ter um processo que reproduz o som de um aplicativo de música e eu movo meu mouse, teoricamente meu mouse se mover é mais importante que a música ser executada, logo o SO troca o processo de execução de som pelo processo de atualizar a posição do mouse na tela, considerando que ele tem uma prioridade maior.

9. Um programa é o conjunto de instruções, já um processo é a execução dessas instruções. Citando o livro base da disciplina, "Um processo é apenas uma instância de um programa em execução, incluindo os valores atuais do contador do programa, registradores e variáveis.". Imagino que num contexto de um sistema computacional sem SO, ambos os termos seriam equivalentes, por exemplo, no primeiro código do sensor mostrado no item 2, a main seria o processo e o programa do sistema computacional. Mas considerando um sistema operacional com escalonador e gerenciamento de processos, um programa é a totalidade das instruções a serem realizadas e o processo é a instância atual do programa no SO, por exemplo, no editor de texto do *word* temos um programa principal e dentro dele vários processos que são executados separadamente, como correção de texto, atualização da tela com base no teclado e auto-save, todos esses itens são processos de um programa.