Relatório Sobre a Disciplina de Sistemas Operacionais

William Cardoso Barbosa

[1] Universidade Federal de Rondônia (UNIR)

[2]Departamento de Ciência da Computação

1. Informações Gerais

Este relatório tem por finalidade sistematizar, de forma resumida, o conhecimento sobre algus tópicos abordados ao longo da disciplina de **Sistemas Operacionais**. Isso garante uma fixação de assuntos importantes, corroborando para um maior entendimento prático, o que a disciplina tem de dificuldade para oferecer, tendo em vista que sua maior parte de conteúdo é teórica.

2. Tópico de Threads

Um modelo computacional importante para o entendimento dos **Sistemas Operacionais** são as Threads. Isso foi abordado com excelência durante as aulas. Buscando assuntos diversificados e complexos sobre essa ”ferramenta”.

Essa ferramenta pode ser entendida como um pequeno programa que trabalha como um subsistema, seria uma forma de um processo computacional ser subdividido em duas - tendo em vista que um processo possui uma thread principal - ou mais tarefas. Em outras palavras, podemos denotar thread como sendo um encadeamento de execução. Segue a imagem abaixo:

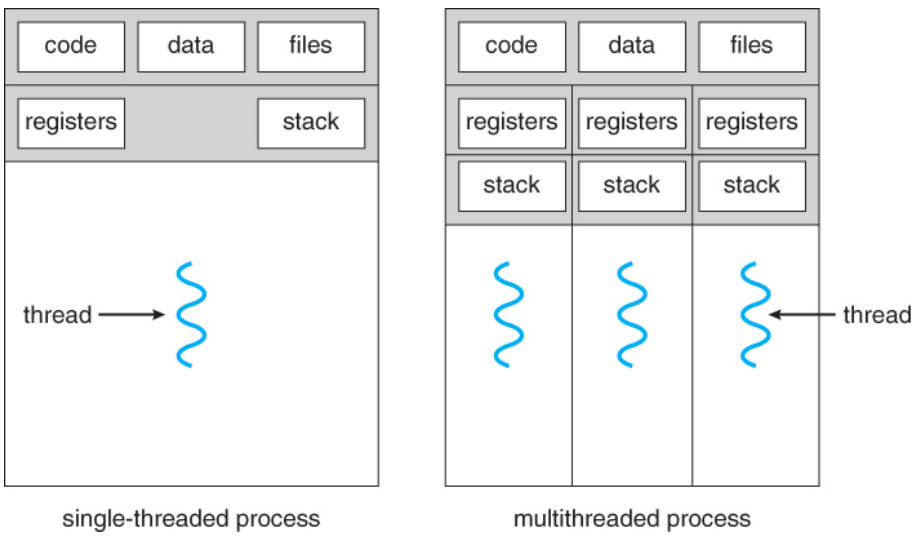

Figura 1. Exemplos da existência de uma thread dentro de um programa

Em adição, as threads que existem em um programa podem trocar dados e informações entre elas e compartilhar os recursos postos a elas durante o funcionamento do programa, isso também inclui a mesma célula de memória. Desse modo, um usuário pode usar outras ferramentas do sistema enquanto as threads trabalham de forma oculta no

sistema operacional. Podemos ilustrar esse cenário da seguinte maneira: Gabriel é usuário do Linux e ele permitiu que seu computador pudesse ser acessado de forma remota via ssh. Gabriel precisa criar 20 pastas em seu computador para organizar as disciplinas de sua faculdade como, por exemplo, pastas "Geometria Análitica", "Sistemas Operacionais", etc. William, um certo dia, decidiu se oferecer para facilitar essa atividade para seu amigo. De forma genial, Gabriel deu a ideia de passar seu usuário ssh para William e assim o convidado possa criar as pastas. A forma mais interessante de fazer essa atividade seria William e Gabriel trabalharem juntos na criação das pastas, com a condição de ambos não criarem uma pasta com o mesmo nome ao mesmo tempo.

Seguindo o problema anterior, o William e Gabriel seriam duas threads em funcionamento, criando diversas pastas - tarefa a ser processada - e o ssh seria a porta que permitiria essa ocorrência. O espaço de memória compartilhado seria a pasta raiz, na qual as outras subpastas seriam criadas. Logo, conseguimos ver de forma macro como as threads funcionam em prol de uma atividade.

Embora isso pareça encantador, as threads possuem desvantagens também, uma delas é que o trabalho fica mais complexo com a quantidade de threads e isso surge justamente por causa da interação que ocorre entre elas. Vimos isso quando impomos uma condicional para todo orquestramento da criação das pastas, mesmo que 2 pessoas estejam criando pastas de locais diferentes sem se comunicar, ambas podem colidirem, fazendo uma mesma sub-atividade, que seria criar pastas com os mesmo nomes.

Também, é importante distinguirmos a diferenciação de processos e threads, ambos parecem iguais, entretanto um é a unidade do outro. O que melhor distingue uma thread de um processo é o espaço de endereçamento. Todas as threads de um processo trabalham no mesmo espaço de endereçamento, que é a memória lógica do "processo hospedeiro". Isto é, quando se tem um conjunto de threads dentro de um processo, todas as threads executam o código do processo e compartilham as suas variáveis. Por outro lado, quando se tem um conjunto de processos, cada processo trabalha num espaço de endereçamento próprio, com um conjunto separado de variáveis.

3. Estrutura do Sistema Operacional

O tópico de Estrutura do Sistema Operacional foi uma preparação para o entendimento do funcionamento mais complexo dessa ferramenta. Vimos conceitos como **Rotinas, System Calls, Arquiteturas, etc...** Entende-se isso como uma base para pensarmos em um sistema operacional, tendo em vista que isso é mund mais complexo e recheado de ferramentas com conceitos super sofisticados.

Sabemos que o sistema operacional é um conjunto de rotinas que oferece serviços ao usuário e às aplicações mas ocorre que essa é uma forma muito simples de entender um sistema operacional.Na verdade, o conjunto de rotinas é o que chama-se de **núcleo do sistema operacional ou kernel**. Contudo os sistemas operacionais são formados por algo além do núcleo. Os sistemas operacionais possuem também uma linguagem de comandos e diversos utilitários de apoio que são usados para complementar o sistema operacional.

Os usuários e aplicações se comunicam com o núcleo do sistema operacional de maneiras distintas. Em geral, os usuários usam aplicações que invocam as rotinas do sistema operacional, por exemplo, quando gravamos um arquivo. Além disso, os usuários

também usam a linguagem de comandos do sistema operacional para executar tarefas no sistema operacional e também usam programas utilitários para tarefas mais complexas como compilação, transmissão de arquivos por exemplo. O ponto é que cada sistema operacional possui sua própria linguagem de comandos, seus próprios utilitários e demais componentes que formam sua estrutura. Podemos observar a diagramação de um Sistema Operacional (generalização).

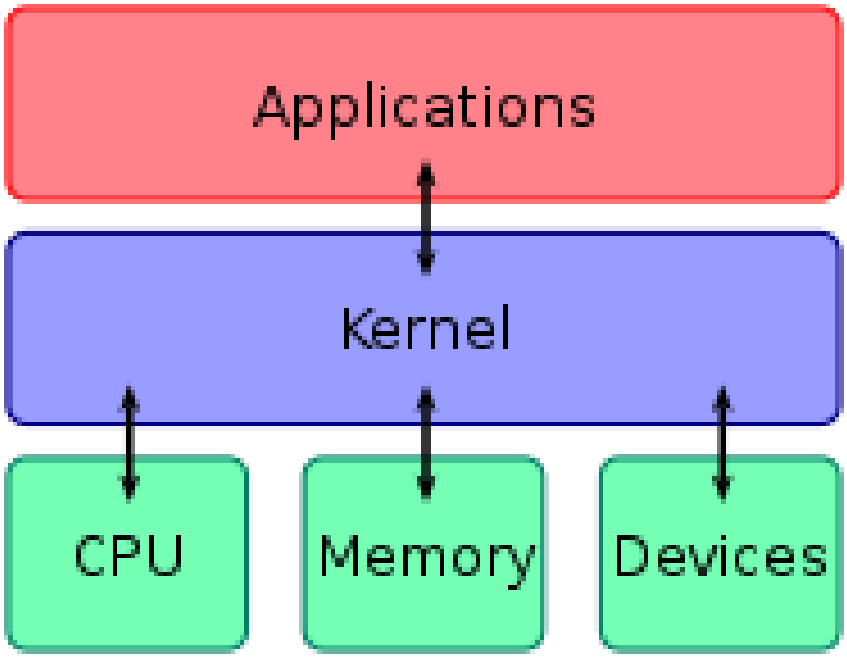

Figura 2. Exemplos da existência de uma thread dentro de um programa

Podemos começar entendo a função do núcleo do sistema operacional porque diferente das rotinas de um programa do usuário, que são executadas segundo uma determinada ordem definida no programa, as rotinas do sistema operacional são executadas concorrentemente sem uma ordem predefinida. A execução dessas rotinas do sistema operacional ocorre com base em eventos assíncronos. Muitos desses eventos são gerados por hardware e por tarefas internas do sistema operacional.

Algumas das principais funções:

- tratamento de interrupções e exceções;
- criação e eliminação de processos e threads;
- sincronização e comunicação entre processos e threads;
- gerencia de memória;
- gerencia do sistema de arquivos;
- gerencia de dispositivos de entrada e saída;
- suporte a redes locais e distribuídas;
- segurança e auditoria do sistema;
- contabilização de uso do sistema;

As rotinas do sistema operacional compõem o núcleo do sistema e possuem em seu código instruções privilegiadas. Portanto, estas rotinas somente podem ser executadas

quando o processador está em modo kernel. Assim, as rotinas do sistema operacional não estão disponíveis para aplicações do usuário e deve ser implementado mecanismo de proteção a tais rotinas.

O mecanismo usa o controle de execução das rotinas do sistema operacional é conhecido como system call. Assim, toda a vez que uma aplicação do usuário chamar uma rotina do sistema operacional o mecanismo de system call é ativado e ele verifica se a aplicação do usuário possui os privilégios necessários para executar a rotina desejada.

Em caso negativo, o sistema operacional impede a execução da rotina e sinaliza que a execução não é possível.

Em caso positivo, o sistema operacional salva o estado dos registradores, troca o modo de acesso para kernel e realiza a execução da rotina do sistema operacional, alterando o registrador PC com o endereço da rotina chamada. Ao termino da execução da rotina, o modo de acesso é alterado para modo usuário.