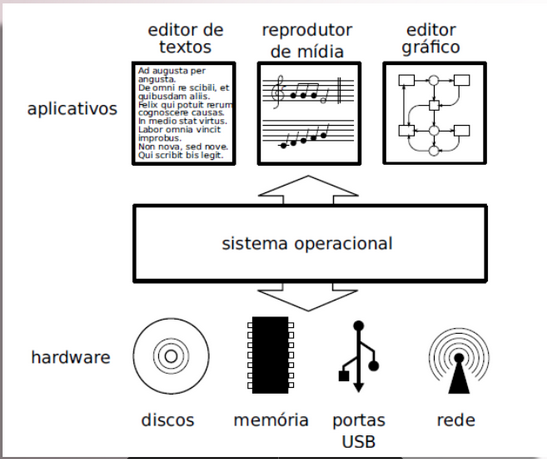
=======================================================

***História dos Sistemas Operacionais***

=======================================================

Os primeiros computadores eram construídos para executar uma tarefa específica, caso você precisasse de realizar outra atividade, um novo projeto deveria ser executado e implementado.

Em uma evolução, o usuário introduzia programa na máquina, introduzia os dados, iniciava o programa e aguardava a finalização.

A partir do advento dos sistemas operacionais muitas possibilidades foram expandidas.

No início, na era dos grandes mainframes, era necessário carregar o computador com toda a programação que iria utilizar.

O primeiro sistema operacional realmente ativo foi o GM-NAA I/O, da GM, criado para o IBM 704.

A proposta era automatizar diversas atividades sem ter que reenviar constantemente códigos para fazer uma determinada função, além de gerenciar melhor os recursos do equipamento.

Com o passar dos anos, diversas empresas projetaram e desenvolveram S.O. para seus mainframes, determinadas empresas se especializaram na criação de sistemas para terceiros.

O problema é que cada um destes sistemas apresentava características próprias e não possuíam nenhum tipo de comunicação entre si, o que criou a necessidade de uma padronização.



Portanto, torna-se desejável oferecer aos programas aplicativos uma forma de acesso homogênea aos dispositivos físicos, que permita abstrair as diferenças tecnológicas entre eles.

Os objetivos básicos de um sistema operacional podem ser sintetizados em duas palavras-chave: “ABSTRAÇÃO” e “GERÊNCIA”, cujos principais aspectos são detalhados a seguir.

=======================================================

***Abstração de Recursos***

=======================================================

Como visto anteriormente, acessar os recursos de hardware de um sistema de computação pode ser uma tarefa complexa, devido às características específicas de cada dispositivo físico e a complexidade de suas interfaces.

Assim, o sistema operacional deve definir interfaces abstratas para os recursos do hardware, visando atender os seguintes objetivos:

* Prover interfaces de acesso aos dispositivos;
* Tornar os aplicativos independentes do hardware;
* Definir interfaces de acesso homogêneas para dispositivos com tecnologias distintas.

=======================================================

***Gerencia de Recursos***

=======================================================



Em um sistema com várias atividades simultâneas, podem surgir conflitos no uso do hardware, quando dois ou mais aplicativos precisam dos mesmos recursos para poder executar.

Cabe ao sistema operacional definir políticas para gerenciar o uso dos recursos de hardware pelos aplicativos.

Cada computador normalmente possui menos processadores que o número de tarefas em execução. Por isso, o uso desses processadores deve ser distribuído entre os aplicativos presentes no sistema.

Assim, um sistema operacional visa abstrair o acesso e gerenciar os recursos de hardware, provendo aos aplicativos um ambiente de execução abstrato, no qual o acesso aos recursos se faz através de interfaces simples.

Para cumprir seus objetivos de abstração e gerência, o sistema operacional deve atuar em várias frentes.

=======================================================

***Gerência do processador***

=======================================================

Também conhecida como ***Gerência de Processos*** ou de ***Atividades***, esta funcionalidade visa distribuir a capacidade de processamento de forma justa entre as aplicações, evitando que uma aplicação monopolize esse recurso e respeitando as prioridades dos usuários.

=======================================================

***Batch (de lote)***

=======================================================

Os sistemas operacionais mais antigos trabalhavam “por lote”, ou seja, todos os programas a executar eram colocados em uma fila, com seus dados e demais informações para a execução.

O processador recebia os programas e os processava sem interagir com os usuários, o que permitia um alto grau de utilização do sistema.

=======================================================

***De Rede***

=======================================================

Um sistema operacional de rede deve possuir suporte à operação em rede, ou seja, a capacidade de oferecer às aplicações locais recursos que estejam localizados em outros computadores da rede, como arquivos e impressoras.

=======================================================

***Distribuído***

=======================================================

Em um sistema operacional distribuído, os recursos de cada máquina estão disponíveis globalmente, de forma transparente aos usuários. Ao lançar uma aplicação, o usuário interage com sua janela, mas não sabe onde ela está executando ou armazenando seus arquivos: o sistema é quem decide, de forma transparente.

=======================================================

***Multiusuário***

=======================================================

Um sistema operacional multiusuário deve suportar a identificação do “dono” de cada recurso dentro do sistema (arquivos, processos, áreas de memória, conexões de rede) e impor regras de controle de acesso para impedir o uso desses recursos por usuários não autorizados.

=======================================================

***Servidor***

=======================================================

Um sistema operacional servidor deve permitir a gestão eficiente de grandes quantidades de recursos (disco, memória, processadores), impondo prioridades e limites sobre o uso dos recursos pelos usuários e seus aplicativos.

=======================================================

***Embarcado***

=======================================================

Um sistema operacional é dito embarcado (embutido ou embedded) quando é construído para operar sobre um hardware com poucos recursos de processamento, armazenamento e energia. Aplicações típicas desse tipo de sistema aparecem em telefones celulares, sistemas de automação industrial e controladores automotivos, equipamentos eletrônicos de uso doméstico (leitores de DVD, TVs, fornos de microondas, centrais de alarme, etc.).

=======================================================

***Tempo Real***

=======================================================

Ao contrário da concepção usual, um sistema operacional de tempo real não precisa ser necessariamente ultrarrápido; sua característica essencial é ter um comportamento temporal previsível (ou seja, seu tempo de resposta deve ser conhecido no melhor e pior caso de operação).

=======================================================

***Estrutura de um Sistema Operacional***

=======================================================

Um sistema operacional não é um bloco único e fechado de software executando sobre o hardware.

Na verdade, ele é composto de diversos componentes com objetivos e funcionalidades complementares.

Alguns dos componentes mais relevantes de um sistema operacional típico são:

=======================================================

***Núcleo***

=======================================================

É o centro do sistema operacional, responsável pela gerência dos recursos do hardware usados pelas aplicações. Ele também implementa as principais abstrações utilizadas pelos programas aplicativos.

=======================================================

***Drivers***

=======================================================

São códigos específicos para acessar os dispositivos (hardware). Existe um driver para cada tipo de dispositivo, como discos rígidos IDE, SCSI, portas USB, placas de vídeo, etc.

Muitas vezes o driver é construído pelo próprio fabricante do hardware e fornecido em forma compilada (em linguagem de máquina) para ser acoplado ao restante do sistema operacional.

=======================================================

***Código de Inicialização***

=======================================================

A inicialização do hardware requer uma série de tarefas complexas, como reconhecer os dispositivos instalados, testá-los e configurá-los adequadamente para seu uso posterior.

Outra tarefa importante é carregar o núcleo do sistema operacional em memória e iniciar sua execução.

=======================================================

***Programas Utilitários***

=======================================================

São programas que facilitam o uso do sistema computacional, fornecendo funcionalidades complementares ao núcleo, como formatação de discos e mídias, configuração de dispositivos, manipulação de arquivos (mover, copiar, apagar), interpretador de comandos, terminal, interface gráfica, gerência de janelas, etc.

=======================================================

***Processos***

=======================================================

O conjunto dos recursos alocados a uma tarefa para sua execução é denominado processo.

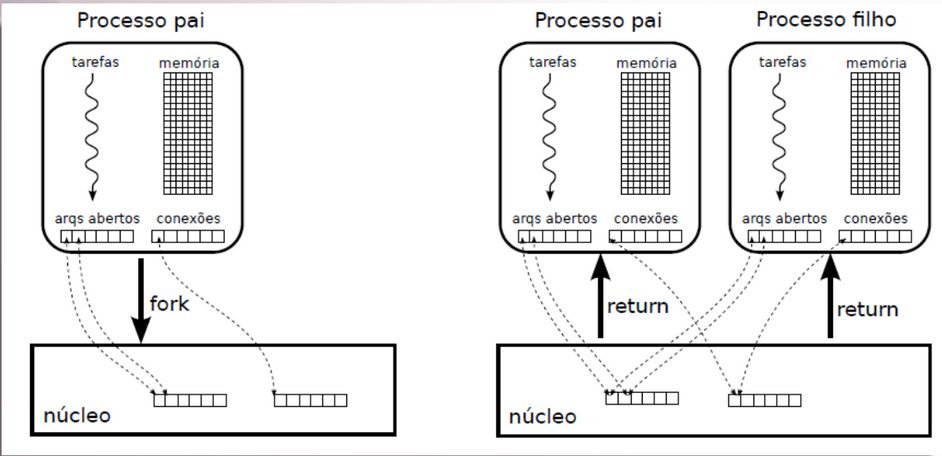
Além do código executável, cada tarefa ativa em um sistema de computação precisa de um conjunto de recursos para executar e cumprir seu objetivo.

Entre esses recursos estão:

* As áreas de memória usadas pela tarefa para armazenar seu código;
* Dados;
* Pilha,
* Arquivos abertos,
* Conexões de rede, etc.

=======================================================

***Criação de Processos***

=======================================================

Durante a vida do sistema, processos são criados e destruídos.

Essas operações são disponibilizadas às aplicações através de chamadas de sistema; cada sistema operacional tem suas próprias chamadas para a criação e remoção de processos.

No caso do UNIX, processos são criados através da chamada de sistema fork, que cria uma réplica do processo solicitante: todo o espaço de memória do processo é replicado, incluindo o código da(s) tarefa(s) associada(s) e os descritores dos arquivos e demais recursos associados ao mesmo.

A chamada de sistema fork é invocada por um processo (o pai), mas dois processos recebem seu retorno: o processo pai, que a invocou, e o processo filho, recém criado, que possui o mesmo estado interno que o pai (ele também está aguardando o retorno da chamada de sistema).

Ambos os processos têm os mesmos recursos associados, embora em áreas de memória distintas. Caso o processo filho deseje abandonar o fluxo de execução herdado do processo pai e executar outro código, poderá fazê-lo através da chamada de sistema execve..

=======================================================

***Threads***

=======================================================

<https://youtu.be/xNBMNKjpJzM?t=114> a partir do 1:54 até 5:02

Os primeiros sistemas operacionais suportavam apenas uma tarefa por processo. À medida em que as aplicações se tornavam mais complexas, essa limitação se tornou um claro inconveniente.

Por exemplo, um editor de textos geralmente executa tarefas simultâneas de edição, formatação, paginação e verificação ortográfica sobre a mesma massa de dados (o texto sob edição).

De forma geral, cada fluxo de execução do sistema, seja associado a um processo ou no interior do núcleo, é denominado thread. Threads executando dentro de um processo são chamados de threads de usuário (user-level threads ou simplesmente user threads).

Cada thread de usuário corresponde a uma tarefa a ser executada dentro de um processo.

Os fluxos de execução reconhecidos e gerenciados pelo núcleo do sistema operacional são chamados de threads de núcleo.