

Sistemas Operacionais

Fundamentos da Informática

Prof.º Lúcio Ramos

1º Módulo – Técnico em desenvolvimento de sistemas.

ETEC Prof.ª Ilza Nascimento Pintus.

Alunos:

Eduarda Satiro de Oliveira;

Gisele Santos Ribeiro;

João Pedro Carvalho dos Santos;

Ryan Roberto da Silva.

**Introdução**

A tecnologia está cada mais globalizada e presente no cotidiano, facilitando as tarefas do dia a dia, seja pelo uso de smartphones, tablets, smartwatches ou computadores. Contudo, isso só foi possível com o advento do sistema operacional (SO), uma vez que é ele que tem a função de gerenciar e potencializar os recursos da máquina, tornando possível o seu uso e funcionamento. Sua história começou na década de 1950, época em que os computadores não trabalhavam com o conceito de sistema operacional propriamente dito, visto que as operações eram configuradas através do próprio hardware do computador. Nesse período, era muito comum que uma mesma pessoa projetasse, programasse e utilizasse os computadores. A principal implicação desta abordagem é o fato de que era muito difícil criar rotinas programáveis, exigindo trabalho intenso dos operadores de máquinas e constante intervenção humana para resolver problemas técnicos que apareciam de modo frequente.

Tendo em vista esse problema, o conceito de sistema operacional apareceu durante a segunda geração da computação moderna (1955 - 1965), através da programação em **Batch**. Assim, vários comandos já poderiam ser executados em sequência através de cartões perfurados, eliminando parte do trabalho do operador de terminal. Normalmente, um programa era composto por um conjunto de cartões inseridos pelo usuário do sistema, na ordem correta. Em meados da década de 60, os primeiros sistemas operacionais foram desenvolvidos conforme a evolução da tecnologia da época. Entretanto, cada máquina possuía seu próprio SO específico, o que implicava na incompatibilidade de mainframes distintos. Um dos maiores representantes foi o CTSS, criado pela MIT, sendo lançado em 1961 para o computador IBM 7090.

Visando resolver o problema da incompatibilidade de SOs de máquinas distintas, em 1965, os desenvolvedores Kenneth Thompson, Denny Ritchie e outros dos Laboratórios Bell da AT&T, juntamente com a General Electric e o MIT (Instituto de Tecnologia de Massachusetts), desenvolveram um protótipo de um sistema operacional chamado **Multics**. Entretanto, o projeto fracassou devido às limitações tecnológicas da época e diferenças entre os grupos envolvidos na pesquisa. Contundo, em 1969, Ken Thompson começou a reescrever o Multics em um conceito menos ambicioso, batizado de **Unics**, usando a linguagem de montagem assembly. Logo depois, o sistema recebeu o nome de **Unix**.

Em 1973, com ajuda de Dennis Ritchie, o Unix foi reescrito novamente, mas dessa vez para a linguagem de programação C, criada pelo próprio Ritchie para substituir e superar as limitações da linguagem utilizada por Thompson. O projeto teve grande sucesso, sendo o primeiro sistema operacional moderno da história, introduzindo conceitos importantes na computação, como portabilidade, multiusuário, multitarefas e compartilhamento de tarefas, ou seja, a partir daquele momento, os computadores poderiam executar dezenas de processos de modo simultâneo e independente e por usuários diferentes. Sua interface era totalmente em modo texto, sem interface gráfica. Durante a década de 70, o Unix foi distribuído gratuitamente (incluindo seu código fonte) para universidades e órgãos governamentais norte-americanos, o que conferiu muita popularidade a este sistema, como também, promoveu, o surgimento de outros sistemas operacionais, como o **BSD** (Berkeley Software Distribution), focado principalmente para a execução em máquinas específicas de alto desempenho, como o famoso computador VAX, o qual foi uma referência de hardware na época.

Tanto o Unix quanto o BSD, em suas primeiras versões, foram desenvolvidos para o uso de computadores de grande porte, normalmente em universidades. Contudo, alguns jovens programadores possuíam uma ideia inovadora para época: criar sistemas operacionais para o uso de pessoas comuns. Um dos primeiros a pensar desta forma foi Steve Jobs, fundador da Apple. Desde a criação de sua empresa, seu principal foco foi a criação de computadores para o dia a dia, incluindo sistemas operacionais fáceis de serem operados. O lançamento do Apple I em 1976, um dos primeiros computadores pessoais (PC), foi um marco na história da computação. Pela primeira vez, um PC continha um teclado fácil de ser utilizado, com uma minitelevisão adaptada como monitor. Assim, conhecimentos avançados de computação já não eram mais requisitos para se operar um computador. Jobs fez questão de criar o seu sistema operacional, do zero, sem se basear inicialmente no Unix. Dessa forma, nos anos seguintes, os sistemas operacionais **Apple DOS** e **Apple SOS** foram desenvolvidos para os computadores Apple II e Apple III, respectivamente. Esses sistemas funcionavam em discos (época em que não existiam os HDs) e suas interfaces era bem rudimentar e contavam apenas com alguns componentes básicos: um gerenciador de arquivos, um catálogo, funções para abrir e remover dados, um programa de inicialização e alguns outros elementos, porém, foi um grande avanço para a época.

Ao final da década de 70, outro jovem programador, chamado Bill Gates, também possuía o desejo de revolucionar o mundo da informática. Em 1975, ele fundou a Microsoft, empresa que possuía como objetivo primário o desenvolvimento de software em linguagem BASIC para o computador Altair da IBM. Com o sucesso dos programas desenvolvidos pela Microsoft, a empresa afirmou que possuía um sistema operacional completo. A IBM se interessou pelo projeto e ambas as organizações afirmaram um contrato, em 1979. Porém, a Microsoft estava com sérios problemas, pois não possuía um sistema operacional de verdade. A solução encontrada foi a compra do SO da Seattle Computer Products, o **QDOS** (Quick and Dirty Operating System), pelo valor de $50.000. Ele foi licenciado para a IBM (para a utilização no IBM PC) como **MS-DOS** (Microsoft Disk Operating System) ainda no ano de 1981 e se tornou o sistema de base para a maioria dos computadores pessoais posteriores. Fechando o acordo com a IBM, a única exigência da Microsoft era que os direitos de revenda do MS-DOS pudesse ser dela. A IBM não se preocupou com isso, afinal, na época, cada sistema operacional era único para um computador, por questão de arquitetura.

Em meados de 1979, Steve Jobs tomou conhecimento sobre o desenvolvimento de um computador totalmente inovador pela Xerox Parc. Em uma visita a esta empresa, ele ficou deslumbrado com Xerox Alto, um PC que possuía um sistema operacional com interface gráfica (GUI) totalmente revolucionária chamada de **Xerox Star**. Pouco tempo depois, a Apple lançou o computador Lisa, com o sistema operacional **Lisa OS**, que aproveitava todas as ideias gráficas do SO da Xerox, Logo depois, foi desenvolvido o sistema operacional **MAC OS** para o Apple Macintosh, lançado em 1984, introduzindo o conceito de desktop (área de trabalho), utilizando ícones e pastas para representar programas e arquivos do modo como se conhece hoje, simplificando e facilitando bastante a usabilidade, dessa forma, dando início a era dos computadores “amigáveis”.

Neste mesmo ano, um revolucionário programador chamado Richard Stallman começou a desenvolver o **Projeto GNU** (uma abreviação recursiva em inglês, que significa "GNU Não é Unix"). Stallman afirmava que os softwares deveriam ser desenvolvidos de maneira livre, sem restrições na leitura e com permissões para modificação nos seus códigos fontes, surgindo então a Filosofia de Software Livre: qualquer pessoa pode copiar e redistribuir cópias, modificar o código, aprimorá-lo e compartilhá-lo com a comunidade, além de poder estudar o funcionamento do programa e executá-lo para qualquer fim. Um dos principais objetivos da GNU sempre foi desenvolver a sua própria versão do Unix, através de um Kernel (núcleo de um SO) próprio, chamado de **GNU Hurd**. Contudo, este núcleo possuía muitas falhas de sistema, comprometendo muito o seu desenvolvimento.

Usando o BSD como base, a Sun Microsystems criou seu próprio sistema operacional, em 1982, chamado de **SunOS**. O software foi desenvolvido para ser comercializado junto aos servidores e estações de trabalho criados pela companhia. Ele durou um bom tempo e continuou evoluindo, até que a desenvolvedora mudou seu nome para **Solaris** em 2005.

Paralelamente a isso tudo, a Microsoft andava mal com as vendas do MS-DOS, pois sua interface era baseada em modo texto, bastante parecida com a utilizada pelo Unix. Na época, esse SO não chamava tanta atenção, pois o Apple Lisa de Steve Jobs já trabalhava com uma interface gráfica. Tomando conhecimento deste problema, Bill Gates fez uma visita à Apple, com o objetivo básico de conhecer a empresa. Ao final da visita, Gates tinha convencido Jobs a ser incluído no desenvolvimento do Macintosh. A partir dessa parceria, Gates copiou a interface gráfica do MAC OS, fazendo melhorias em alguns dos seus aspectos para então ser utilizada como base para a criação do novo sistema operacional da Microsoft: o **Windows 1.0**, lançado em 1985.

Após problemas de administração, Jobs foi demitido da Apple, o que desestabilizou a empresa, retornando somente em 1997. Assim, a Microsoft foi ganhando cada vez mais espaço no mercado, lançando o **Windows 2.0** em 1987, trazendo melhorias consideráveis na parte visual e no gerenciamento de memória.

No início dos anos 90, o mercado de sistemas operacionais se popularizou bastante com o lançamento do **Windows 3.0** (1990) e **Windows 3.1** (1992). Na sua versão 3.0, a memória passou a ser gerenciada de maneira muito mais eficiente, incluindo a melhora substancial na interface gráfica. Foi criado um painel de controle e um gerenciador de arquivos organizado, facilitando todo o trabalho do usuário. Um dos principais motivos que contribuíram para seu sucesso foi o fato do sistema já vir instalado de fábrica em muitas máquinas, desse modo, popularizando ainda mais os computadores pessoais.

Com o intuito de criar um núcleo de sistema operacional, um programador chamado Linus Torvalds, aproveitando as bibliotecas e aplicações do GNU, em 1991, criou o **Linux**. Desde o começo, Torvalds distribuiu o Kernel de forma gratuita, por isso, foi atribuído a licença GNU ao Linux, criando-se, assim, o sistema operacional **GNU/Linux**, que já possuía mais funcionalidades que o GNU Hurd, o que atraiu bastantes desenvolvedores. Pouco tempo depois, devido a tanto sucesso, recebeu várias contribuições de programadores ao redor do mundo, garantindo que diversos outros sistemas pudessem ser desenvolvidos e oferecidos com novas experiências para os usuários. Essas novas versões adicionavam sistemas de janelas, compatibilidade com outros sistemas de arquivos, bibliotecas para determinadas funções, pacotes exclusivos etc. Grandes empresas também contribuíram na programação de seu código ao longo da história como IBM, Sun Microsystems, Hewlett-Packard (HP), Red Hat, Novell, Oracle, Google, Mandriva e Canonical, e atualmente o Linux é o kernel mais utilizado, desde computadores de grande porte, passando por computadores pessoais, DVD player, roteadores e celulares. As versões do Linux para desktop mais conhecidas hoje são: Ubuntu, Solus e o Debian.

No ano de 1995, foi lançada no mercado uma nova versão do Windows, o **Windows 95**. Esta versão foi tão importante para informática que acabou definindo o padrão com que o desktop é organizado até hoje, como, por exemplo, o botão Start, o menu Iniciar, a barra de tarefas e o gerenciador de arquivos Windows Explorer. Após algumas atualizações, esta versão passou a suportar a leitura de dispositivos USB, o navegador internet explorer, entre outras funcionalidades que facilitaram a utilização do SO. Esta versão foi a responsável por tornar o Windows o sistema mais usado do mundo.

Em 1998, substituindo o Windows 95, a nova versão do sistema, chamada de **Windows 98**, apresentou a capacidade de rodar programas de 16 e 32 bits. Seus principais diferenciais eram o suporte já nativo para dispositivos USB e para drivers mais avançados. Além disso, ele vinha com o DirectX 5, que garantia a execução de jogos mais pesados. Apesar de apresentar melhorias em relação ao 95, o SO era um pouco lento e instável. Tais problemas só foram arrumados com o **Windows 98 SE** (Second Edition), lançado em 1999, que incluía funções avançadas para compartilhamento de rede, suporte integrado a drivers de DVD-ROM, entre outras tarefas.

O sucessor, **Windows Me**, lançado em 2000, foi um dos maiores fracassos na questão de sistema operacional, pois era muita instável. Possuía somente poucas melhoras em relação ao Windows 98 SE e foi o sistema com a maior quantidade de bugs que a Microsoft já lançou. Além de ser lento, ele focava apenas em alguns recursos visuais que não ajudaram o sistema a ficar vivo por muito tempo. Por isso, logo foi deixado de lado.

Todas as versões apresentadas até aqui usavam o MS-DOS como núcleo do sistema, ou seja, o Windows funcionava como uma espécie de ambiente gráfico. Com o passar do tempo, o uso desta arquitetura tornou-se insuportável, visto que o MS-DOS não conseguia dar conta de processar tantas informações, o que ficou evidente no Windows ME.

Em 2001, o lançamento do **Windows XP** tornou-se um marco na história dos sistemas operacionais, principalmente por trazer muitos recursos totalmente novos. Ele teve sua interface completamente remodelada, trouxe uma série de novos recursos para redes, DirectX 8.1, suporte avançado para múltiplos usuários, novos recursos de segurança através da diferenciação de permissões entre administradores e usuários comuns. A estabilidade também é uma de suas fortes características, pois o número de telas azuis diminuiu consideravelmente. Este foi, provavelmente, o SO mais usado nos anos 2000 e ganhou muitas atualizações durante sua vida ativa.

Ainda em 2001, após tantos anos trabalhando em seu sistema próprio, a Apple finalmente lançou um sistema baseado no UNIX. O **OS X** veio para revolucionar a história da companhia. Desde o lançamento desse sistema, a Apple vem apenas realizando melhorias e constantemente lançando novas versões até hoje. O **Mac OS X**, como também era chamado, trazia memória protegida, o dock (semelhante ao que é usado até hoje), o terminal, um cliente de e-mail, suporte para OpenGL e outros tantos recursos.

Depois de seis anos, a Microsoft lança o **Windows Vista**, em 2007, que foi muito aguardado pelos usuários. Ao contrário do XP, esta nova versão desapontou o público de uma maneira geral, principalmente por exigir uma máquina muito potente. Somente com computadores avançados da época era possível observar vantagens no desempenho do Vista, principalmente pelo suporte a multicore. Seu grande destaque foram os efeitos gráficos de última geração provido pelo Aero e o Flip 3D. Apesar disso, alguns recursos (como o Windows Search e o Windows Aero) foram inovações que ajudaram no desenvolvimento do sistema sucessor.

Na segunda metade dos anos 2000, com o avanço das plataformas móveis, diversos sistemas operacionais com particularidades diferentes, específicos para esses equipamentos, foram criados. Eles precisavam ser mais leves e continham uma lógica de interação completamente diferente.

Nessa época foi quando surgiu o **Symbian** e o **BB**, da Nokia e Blackberry, respectivamente. Esses sistemas foram responsáveis pela popularização dos smartphones, trazendo acesso a aplicativos de produtividade e segurança modernos a usuários mais leigos. Porém, desde os anos 90 havia sistemas operacionais para computadores de mão (PDA), como o **palmOS** e o **NewtonOS**, mas nem de longe conseguiram a popularização dos smartphones.

No mesmo ano, ocorreu a grande revolução no mercado de celulares quando a Apple lançou o famoso iPhone com seu sistema operacional **IOS**. Na época, não havia nada semelhante ao smartphone e sistema da Maçã. O IOS foi criado com base no OS X e introduziu ao mundo recursos de fácil uso, como teclado virtual, conexão EDGE, Wi-Fi, apps e e-mail. Até aquele momento, o aparelho era utilizado apenas para enviar mensagens de texto e fazer ligações. Recentemente, o sistema passou por uma reformulação em seu visual e continua dando passos largos na introdução de novos recursos.

Em 2008 a gigante Google lança no mercado o Sistema Operacional **Android**, com seu núcleo em Linux. Anteriormente desenvolvido pela Android Inc., o [Android](https://www.tecmundo.com.br/android) foi adquirido pela Google ainda em 2005. De lá para cá, a gigante das buscas veio trabalhando para conquistar o mercado móvel. Hoje, o Android é o sistema portátil mais usado do mundo, sendo que ele vem ganhando destaque por ser pioneiro no lançamento de diversos recursos, como navegador Web, Animações de transições de tela, suporte para Widgets e upload de vídeos para o [YouTube](http://www.techtudo.com.br/tudo-sobre/youtube-fabricante.html) e imagens para o [Picasa](http://www.techtudo.com.br/tudo-sobre/picasa.html).

Considerando as decepções do Windows Vista, a Microsoft novamente resolveu correr atrás do prejuízo. O **Windows 7** vem para corrigir todos os problemas do antecessor e oferecer desempenho acima de tudo. O sistema não alcançou a apreciação que o Windows XP teve, mas conseguiu conquistar o público.

Depois de insistir longos anos no **Windows Mobile** (que era apenas uma versão mal adaptada do sistema para desktop), a Microsoft remodelou seu sistema para portáteis. O Windows Phone conta com ícones grandes, tiles, um design muito limpo e compatibilidade com APIs modernas para a execução de jogos tridimensionais.

Pensando em migrar para o setor dos tablets e revolucionar a forma como as pessoas usam os PCs, a Microsoft apostou em uma reformulação de seu sistema. Aproveitando elementos do Windows Phone, o novo [**Windows 8**](https://www.tecmundo.com.br/windows-8), lançado em 2012, trouxe um novo Menu Iniciar. Além disso, ele mantém a compatibilidade com os antigos programas e traz suporte para novos apps.

3 anos depois, em 2015, a Microsoft lança o **Windows 10**. O principal destaque desta versão foi a reversão da interface para o tradicional paradigma desktop do Windows 7, com Barra de Tarefas e Menu Iniciar. Recursos do Windows 8 como os apps modernos e a loja de aplicativos persistem, mas de forma muito mais integrada à interface tradicional, e muito mais familiar aos usuários de longa data. A evolução do Windows 10 seguiu constante, logo, a Microsoft não tratava mais seu sistema operacional como um produto, mas como um serviço. Duas vezes ao ano, a marca liberou pacotes de atualizações massivas para renovar a experiência (do visual ao funcional) e muita coisa mudou desde o lançamento.

Hoje, o **Windows 11** é a atual versão do Windows, apresentada ao público no final de 2021. Esta versão inclui um novo visual, funcionalidades e uma abordagem mais amigável com o Linux (incluindo [suporte a apps de Android](https://canaltech.com.br/apps/windows-11-tera-suporte-nativo-a-aplicativos-de-android-188172/)) e muito mais.

**Definição**

O sistema operacional um software ou conjunto de softwares cuja função é administrar e gerenciar os recursos de um sistema, desde componentes de hardware e sistemas de arquivos a programas de terceiros, estabelecendo a interface entre o computador e o usuário.

O sistema operacional introduz uma “camada de abstração” entre o hardware e o usuário, que transforma comandos no mouse ou teclado e solicitações do sistema, como gerenciamento de recursos (CPU, memória RAM), em linguagem de máquina, enviando instruções ao processador. Este último os traduz para código binário, executa os comandos e envia as respostas como informações que aparecem na tela.

**Estrutura Genérica de um Sistema Operacional:**

**Diagrama

Descrição gerada automaticamente**

**Kernel:**

Um sistema operacional é **formado por várias rotinas**, que em conjunto são denominadas **núcleo do sistema**, ou **kernel**. 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 (simultaneamente) sem uma ordem predefinida. A execução dessas rotinas ocorre com base em eventos assíncronos. Muitos desses eventos são gerados por hardware e por tarefas internas do sistema operacional.

Contudo os sistemas operacionais são formados por algo além do núcleo. Os SOs possuem também uma linguagem de comandos e diversos utilitários de apoio que são usados para complementá-los.

Em geral, as aplicações, quando estão sendo utilizadas pelos usuários, comunicam-se com o kernel, invocando as rotinas do sistema, como, por exemplo, a entrada e saída de dados. Além disso, os usuários podem usar também a linguagem de comandos para executar tarefas no sistema operacional e usar programas utilitários para tarefas mais complexas, como compilação e transmissão de arquivos por exemplo. O ponto é que cada SO possui sua própria linguagem de comandos, seus próprios utilitários e demais componentes que formam sua estrutura.

**Modo de Acesso:**

Em todo projeto de sistema operacional os projetistas sempre se preocuparam em garantir a integridade do próprio SO, protegendo o núcleo e o serviços do sistema em operação de acessos indevidos por uma aplicação do usuário. Caso uma aplicação do usuário tenha acesso ao núcleo e realize alguma operação que altere sua integridade, então todo o sistema ficará comprometido. Muitos sistemas operacionais utilizam um mecanismo de segurança implementado no hardware do processador, conhecido como **modo de acesso**.

Este mecanismo de proteção funciona através do conceito de **modo usuário** e **modo kernel**.

Quando o processador trabalha em **modo usuário**, uma aplicação tem acesso a apenas algumas instruções não-privilegiadas, tendo acesso a um número reduzido de instruções. Quando o processador está em **modo kernel,** a aplicação que está sendo executada tem acesso a todo o conjunto de instruções do processador, ou seja, tem acesso as instruções privilegiadas. O **modo de acesso** é determinado por um conjunto de bits no registrador de status. O hardware verifica este registrador para determinar se uma instrução pode ou não ser executada.

**Rotinas do Sistema Operacional e System Calls:**

As **rotinas do sistema operacional** compõem o núcleo e possuem instruções privilegiadas em seu código. Portanto, estas rotinas somente podem ser executadas quando o processador está em modo kernel. Assim, as rotinas do SO não estão disponíveis para aplicações do usuário e por isso deve-se ser implementado um mecanismo de proteção a tais rotinas.

Tal mecanismo que usa o controle de execução das rotinas do sistema operacional é conhecido como **system call**. Assim, toda vez que uma aplicação do usuário chama uma rotina do SO, 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, alterando o registrador PC com o endereço da rotina chamada. Ao término da execução, o modo de acesso é alterado para modo usuário. O termo system calls é comum em sistemas UNIX. Em outros sistemas existem termos como **system services**.

**Linguagem de Comandos:**

A linguagem de comandos ou linguagem de controle permite que os usuários se comuniquem com o sistema operacional de forma simples através de uma interface de comandos, que são interpretados pelo interpretador de comandos para realização de tarefas, como ler arquivos ou consultar pastas (diretórios). A linguagem de controle pode usar uma interface de linha de comandos ou uma interface gráfica. Para cada comando executado pelo usuário, o interpretador de comandos o reconhece, analisa a sua sintaxe e verifica permissões para a sua execução. Cada sistema operacional possui sua linguagem de controle específica. Os comandos podem ser gravados em arquivos de script para serem executados como se fossem programas. Esta forma de uso é muito usada para a automação de tarefas rotineiras, como backup, carga de dados etc. Um exemplo de interpretador de comandos seria o [Bash](https://pt.wikipedia.org/wiki/Bash), sendo comumente utilizado no Linux e no macOS.

A linguagem de comandos não faz parte do núcleo do sistema. Ela é um componente da arquitetura do SO e pode variar mesmo em versões do mesmo sistema operacional.

**Ativação/Desativação do Sistema:**

Inicialmente quando o computador é ligado não existe sistema operacional carregado na memória da máquina ainda. O sistema operacional é carregado através de um processo gravado na memória ROM da máquina chamado ativação do sistema ou **boot**.

O processo de ativação do sistema é iniciado automaticamente quando o computador é ligado e inicialmente executa um programa chamado **BIOS** (Basic Input/Output System), que é executado a partir da memória ROM.

A BIOS executa um outro programa chamado **POST** (Power On Self Test), que faz uma verificação do hardware da máquina para identificar possíveis problemas de hardware.

Em seguida, a BIOS busca no computador algum dispositivo de armazenamento que tenha um sistema operacional instalado (caso nenhum dispositivo de armazenamento tenha um sistema operacional disponível é exibida uma mensagem de erro e o processo finaliza).

Após isso, a BIOS ativa um programa chamado **bootloader**, ou gerenciador de boot, que busca no dispositivo de armazenamento o setor de boot onde carrega um conjunto de instruções para inicialização do sistema operacional.

A partir dele, é inicializado o núcleo (kernel). Assim como o BIOS estabelece a ligação entre hardware e sistema, o kernel serve para firmar uma comunicação estável entre hardware e software.  Nessa fase, é ele quem assume o controle do computador.

O kernel carrega os arquivos principais e informações básicas do sistema operacional (incluindo o registro), além de relacionar os componentes de hardware com os respectivos drivers.

 No entanto, o kernel não carrega todos os processos para não sobrecarregar o sistema — somente as operações essenciais são colocadas em atividade para possibilitar o início do SO.

 A tela de escolha de usuários é exibida e, após o logon, os programas relacionados para começar junto com o sistema são carregados.

O processo de desativação do SO, também conhecido como **shutdown**, é um processo que é executado a partir de comandos e permite que tantos os componentes do sistema operacional e programas do usuário sejam encerrados de forma a garantir a integridade do sistema.

**Tipos de Sistemas Operacionais**

Sistemas operacionais são divididos em categorias que definem as suas características. Sistemas modernos podem usar combinações de duas ou mais dessas categorias descritas a seguir:

**Sistema Monoprogramado (Monotarefa):** sistema operacional que executa apenas um processo de cada vez. possui apenas um processador, e realiza alocação de memória para somente um processo e um usuário por vez. Sua implementação é feita de forma simples, pois o sistema dedica todos os recursos do computador para somente a execução de uma tarefa. Contudo, mostra-se como uma tremenda desvantagem, visto que a CPU fica ociosa quando não está sendo utilizada pelo programa em execução, enquanto poderia estar sendo utilizada por um outro programa em uma outra tarefa, logo, este modelo possui grande lentidão.

Foram os primeiros SOs utilizados pelos computadores nos anos 60. Foram utilizados também nos microcomputadores e estações de trabalho, lançados nos anos 70 e que eram projetados para uso por apenas um usuário.

Atualmente este tipo de sistema operacional já não é usado mesmo em computadores pessoais, pois com o avanço da tecnologia, aumentou a necessidade de se realizar mais de uma tarefa ao mesmo tempo.

**Sistema Multiprogramado/Multitarefa**: Esse tipo de sistema operacional permite que vários programas sejam executados, compartilhando os recursos do computador, tais como discos, impressora, memória e processador. Neste sistema, enquanto um programa espera uma entrada de dados pelo usuário, um outro programa pode estar sendo processado no mesmo intervalo de tempo. Neste caso tanto a memória quanto o processador são compartilhados e o SO deve gerenciar e controlar este compartilhamento dos recursos de forma segura e protegida. Como existem vários programas sendo executados, é função do sistema operacional garantir que um programa não afete o outro, permitindo a execução das tarefas de forma independente umas das outras.

Por permitir o compartilhamento dos recursos pelas várias tarefas sendo executadas, este tipo de sistema operacional possibilita a redução de custos em hardware. Entretanto, esses sistemas são de implementação muito mais complexa, por ser necessário o gerenciamento e controle de todas as tarefas em execução e do compartilhamento dos recursos entre as tarefas.

Os sistemas multiprogramados podem ser classificados de duas formas:

Pelo número de usuários que interage com o sistema operacional;

Pela forma como as tarefas são gerenciadas;

Com relação ao número de usuários, os sistemas multiprogramados podem ser: **monousuário** ou **multiusuário**. Os monousuários permitem que sejam utilizados apenas por um usuário por vez, apesar de alguns poderem suportar recursos como troca de usuário, assim como o [Windows](https://pt.wikipedia.org/wiki/Windows), que também pode ser acessado por terminais e conexão remota. Os sistemas multiusuários são sistemas acessados por vários usuários que podem interagir com o sistema operacional de forma simultânea. É o caso de sistemas UNIX e LINUX.

Com relação a forma como as tarefas são gerenciadas, os sistemas multiprogramados podem ser:

* Sistemas Bach (em Lote)
* Sistemas de Tempo compartilhado (Time-Sharing)
* Sistemas de Tempo real (real time).

Um sistema operacional multiprogramado pode suportar uma ou mais formas de gerenciamento de tarefa.

**Sistemas multiprogramados Bach** - Foram os primeiros sistemas operacionais multiprogramados e foram utilizados nos anos 60. Neste tipo de sistema, os programas, chamados de Jobs, aguardavam disponibilidade de memória principal para serem executados.

Se houvesse memória principal disponível, mais de um programa era executado. Caso não houvesse, os programas ficavam em uma fila, aguardando a disponibilidade para serem processados.

Neste sistema não havia desperdício de recursos de memória e mais de um programa podia ser executado. Além disso, seu tipo de processamento exige pouca ou nenhuma interação do usuário com os programas. Todas as entradas e saídas de dados são executadas com memória secundária (discos, fitas), o que permite melhor utilização do processador.

**Sistemas multiprogramados de tempo compartilhado** - Permitem a execução de várias tarefas pela divisão de tempo do uso do processador em pequenos intervalos de tempo denominados fatia de tempo (time slice). Os programas em execução se alternam no uso do processador, cada um fazendo uso de sua fatia de tempo. Desta forma, o tempo de uso do processador é compartilhado pelas várias tarefas em execução. Por este motivo, estes sistemas são chamados de sistemas time-sharing (tempo compartilhado).

Quando um programa termina seu intervalo de tempo de uso do processador, ele é interrompido pelo sistema operacional, sendo substituído por outro programa e fica aguardando um novo intervalo de tempo para ter sua execução reiniciada do ponto onde foi interrompido.

**Sistemas multiprogramados de tempo real** - Permitem a execução de várias tarefas de acordo com a prioridade de execução de cada tarefa. Difere dos sistemas de tempo compartilhado, pois neste último o tempo de resposta pode variar um pouco entre a execução de cada tarefa. Nos sistemas de tempo real o tempo de resposta é rigidamente controlado e deve estar dentro de limites de tempo definidos que devem ser obedecidos, caso contrário podem ocorrer problemas irreparáveis as aplicações. Costuma-se dizer que estes sistemas têm tempo de resposta quase instantâneo, por isso são conhecidos por sistemas de tempo real (real time).

Neste modelo não existe fatia de tempo para as tarefas em execução. O mecanismo utilizado para o compartilhamento do processador é o nível de prioridade. Ou seja, uma tarefa permanece em execução pelo processador o tempo que for necessário, até que outra tarefa com maior prioridade seja iniciada. A prioridade de execução das tarefas é determinada pela aplicação e não pelo sistema operacional.

Estes sistemas são bastante utilizados para controlar processos de maneira precisa, como no controle de refinarias, controle de tráfego aéreo, sistemas de defesa, usinas termoelétricas, usinas nucleares, experimentos científicos etc.

**Sistema de Multiprocessamento:** Também chamado de sistema com múltiplos processadores, caracteriza-se por ter duas ou mais CPUs interligadas trabalhando em conjunto para o processamento de dados. Desta forma, é possível a execução de várias tarefas ao mesmo tempo.

Este modelo incorpora todos os recursos dos sistemas multiprogramados/multitarefas e acrescentam novas características e vantagens, tais como escalabilidade, maior disponibilidade e balanceamento de carga.

A Escalabilidade permite aumentar o poder de processamento com a adição de novos recursos a um computador. Nestes sistemas pode-se adicionar novas CPUs para ganho de processamento.

A maior disponibilidade é garantida nestes sistemas, pois em caso de falha de um processador, outros processadores assumem a tarefa, sem que afete as aplicações em execução, garantindo, desta forma, a alta disponibilidade dessas aplicações.

O Balanceamento de carga permite uma melhor distribuição de tarefas entre os diversos processadores disponíveis, melhorando o desempenho geral do sistema.

Estes sistemas são usados para aplicações que exigem intenso processamento de dados, requerendo mais de uma CPU, tais como simulações complexas, processamento de imagens, processamento científico de dados, prospecção de petróleo, pesquisa aeroespacial etc.

Por terem estas características estes sistemas também são conhecidos como multicomputadores.

**Arquiteturas do Núcleo (Kernel)**

O projeto de um SO é realizado de forma a atender a requisitos operacionais de desempenho, portabilidade (capacidade de um sistema ser compilado ou executado em diferentes arquiteturas seja de hardware ou de software), confiabilidade, facilidade de uso e segurança. O projeto do sistema operacional também depende da arquitetura de hardware e do tipo de sistema operacional que se deseja construir.

Os primeiros sistemas operacionais eram escritos em Assembly, linguagem de baixo nível (linguagem próxima à linguagem da máquina), isso garantia que o sistema operacional performasse melhor, mas causava uma dependência muito grande em relação ao hardware da máquina, o que comprometia seriamente a portabilidade. O SO tinha que ser reescrito se precisasse ser instalado em uma máquina com arquitetura e hardware diferente.

Os sistemas operacionais atuais são escritos em linguagens de alto nível (linguagem próxima à linguagem humana), principalmente C e C++, o que garante portabilidade e independência de hardware, porém, o desempenho é inferior aos antigos sistemas operacionais. Para minimizar este problema, os projetistas ainda usam Assembly para os componentes essenciais do sistema operacional, tais como drivers e as rotinas de tratamento de exceção.

A forma como os componentes do sistema operacional é organizada pode variar, sendo que existem várias arquiteturas de sistemas operacionais disponíveis, como:

**Arquitetura Monolítica:** A arquitetura monolítica foi utilizada nos primeiros sistemas operacionais, tais como CP/M, MS-DOS e nas primeiras versões do Linux. Nesta arquitetura os componentes do sistema são compilados em módulos separados e depois linkados em um único programa executável. Os módulos são carregados em memória e interagem entre si. A manutenção deste tipo de sistema é bem difícil.

Diagrama

Descrição gerada automaticamente

**Arquitetura de Camadas:** A arquitetura em camadas surgiu devido à complexidade dos sistemas operacionais na medida em que foram evoluindo. Nesta arquitetura, o sistema operacional é formado por níveis ou camadas, onde as camadas inferiores oferecem serviços às camadas superiores, ou seja, essa arquitetura consiste em níveis sobrepostos, nos quais cada camada tem um conjunto de funções que podem ser utilizadas pelas camadas mais externas. A vantagem deste tipo de arquitetura é a maior organização do sistema, e maior segurança e proteção ao kernel, uma vez que sendo localizado nas camadas mais internas, para acessá-lo, é necessário passar pelas mais externas primeiro.

A desvantagem é que o desempenho do sistema é afetado pela troca de modo de acesso. Quando um aplicativo do usuário solicita um serviço da camada kernel, sendo necessário passar por várias outras camadas (supervisor, executivo), é preciso realizar várias trocas do modo de acesso ao longo do caminho.

A maioria dos sistemas operacionais atuais, tais como Linux e Windows utilizam o modelo de arquitetura em camadas, sendo que estes sistemas implementam apenas duas camadas: modo usuário (camada externa) e modo kernel (camada interna).

Uma imagem contendo Diagrama

Descrição gerada automaticamente

**Arquitetura de Máquina Virtual:** Um sistema operacional normalmente é formado por níveis, onde a camada de nível mais baixo é o hardware. Acima dessa camada temos o sistema operacional que oferece serviços para os aplicativos do usuário. Na arquitetura de máquina virtual existe uma camada intermediária entre o hardware e o SO chamada gerência de máquinas virtuais.

Esta camada cria diversas máquinas virtuais independentes, onde cada uma oferece uma cópia virtual do hardware, incluindo os modos de acesso, interrupções, memória, dispositivos de entrada e saída etc. Como cada máquina virtual (Virtual Machine ou VM) é independente das outras, é possível que cada VM tenha seu próprio sistema operacional e que seus usuários executem suas aplicações como se o computador estivesse dedicado a cada um deles.

Cada máquina virtual é isolada das demais o que proporciona segurança para cada VM. Isto garante também confiabilidade, pois uma VM não pode comprometer o estado das outras VMs. A desvantagem desta arquitetura é a sua grande complexidade. A camada de gerência de máquinas virtuais é responsável por compartilhar e gerenciar os recursos do hardware entre as diversas VMS. Por isso, esta é uma arquitetura altamente complexa.

Diagrama

Descrição gerada automaticamente

**Arquitetura MicroKernel:** A arquitetura MicroKernel ou Cliente-Servidor busca tornar o núcleo do sistema, o kernel, o menor e mais simples possível. Para isso, nesta arquitetura, os serviços do sistema operacional são disponibilizados como serviços. Cada serviço oferece um conjunto de funções como gerência de arquivos, gerência de processos, gerência de memória etc.

Quando uma aplicação do usuário solicita um serviço, é feita uma solicitação ao processo responsável por esse serviço. A aplicação que o solicita é chamada cliente e o processo que responde à solicitação é chamado de servidor. O núcleo do sistema se limita a realizar a comunicação entre cliente e servidor. É, portanto, um núcleo muito mais simples.

Além disso, neste tipo de arquitetura, os processos executam suas funções em modo usuário, ou seja, não tem acesso a instruções privilegiadas, ou seja, não tem acesso aos componentes do sistema. Apenas o núcleo executa em modo kernel. Isto garante que caso haja um erro em um processo, o sistema não ficará completamente comprometido, aumentando a sua disponibilidade (capacidade do sistema de resistir a falhas de hardware, software e energia, com o objetivo de manter os serviços disponibilizados o máximo de tempo possível).

Diagrama

Descrição gerada automaticamente

**Tipos de Interface:**

Os sistemas operacionais fornecem abstração de hardware para que seus recursos possam ser usados de maneira correta e padronizada, mas para ser possível operar um computador, é necessário fornecer também uma interface para que o usuário possa desfrutar dos recursos do sistema. Atualmente as principais interfaces de uso são as seguintes:

**Interface de terminal:** A interface de terminal, também chamada de interface de linha de comando ou "CLI" (Command Line Interface) funciona exclusivamente com o teclado. Os comandos são digitados a partir de um [prompt](https://pt.wikipedia.org/wiki/Prompt) e são interpretados por um [interpretador de comandos](https://pt.wikipedia.org/wiki/Interpretador_de_comandos), conhecidos também por shells, bastante comuns em sistemas padrão POSIX (*portable operating system interface for unix*), que é uma biblioteca de chamada, resultado de uma padronização pensada em portabilidade entre sistemas Unix.

Ele é usado geralmente por usuários com maiores conhecimentos em informática, e para atividades específicas, como o gerenciamento remoto. Além disso, essa interface utiliza poucos recursos de hardware em comparação com a interface gráfica.

Texto

Descrição gerada automaticamente

**Interface textual:** Assim como a interface de terminal, a interface textual também é baseada em texto, porém também tem à disposição um ambiente de trabalho composto por menus, janelas e botões. Esse tipo de interface tinha um uso difundido em aplicações baseadas no [MS-DOS](https://pt.wikipedia.org/wiki/MS-DOS), que, inclusive, nas versões mais recentes contava com um gerenciador de programas e arquivos baseados nesse tipo de interface (o DOS Shell). Atualmente essa interface é muito rara, praticamente restrita a sistemas implementados na [década de 1980](https://pt.wikipedia.org/wiki/D%C3%A9cada_de_1980) e início da [década de 1990](https://pt.wikipedia.org/wiki/D%C3%A9cada_de_1990).

Texto

Descrição gerada automaticamente

**Interface gráfica:** Nesse tipo de [interface](https://pt.wikipedia.org/wiki/Interface_do_utilizador), também chamada GUI (Graphic User Interface) além de menus, janelas e botões também existem figuras, tanto vetoriais quanto fotografias. O usuário interage com esse tipo de interface usando o mouse, podendo também usar o teclado e teclas de atalho, ou então usando toques e gestos em [touchscreens](https://pt.wikipedia.org/wiki/Touchscreen).

A interface gráfica permite atuar com algumas aplicações que seriam impossíveis através da linha de comando puramente, como edição de imagem e vídeo. Acrescentar facilidade de uso e agilidade é o objetivo da interface gráfica, tendo a desvantagem de consumir muito mais memória que interfaces de linha de comando.

Ao contrário das interfaces textuais e de terminal, as interfaces gráficas dependem de um servidor gráfico para funcionar e se comunicar com o sistema, e no caso dos sistemas para desktops e laptops, inclui um gerenciador de janelas em muitos casos, para que seja possível usar mais de um aplicativo na mesma tela.

Em sistemas padrão POSIX é comum existir mais de um ambiente gráfico para o mesmo sistema, podendo ser escolhido a critério do usuário.

Interface gráfica do usuário

Descrição gerada automaticamente

**Interface de voz:** Interfaces de voz, ou VUI (Voice User Interface), são aquelas em que o usuário interage com o sistema por meio de comandos sonoros. Sendo de desenvolvimento relativamente recente, tem sua aplicação em dispositivos adaptados para [cegos](https://pt.wikipedia.org/wiki/Cegueira) e atualmente bastante comum para uso geral em smartphones, tablets e desktops.

Celular com tela ligada

Descrição gerada automaticamente

**Finalidade**

A principal função do sistema operacional é gerenciar os recursos do computador (definir qual programa recebe atenção do [processador](https://pt.wikipedia.org/wiki/Unidade_central_de_processamento), gerenciar memória, criar um [sistema de arquivos](https://pt.wikipedia.org/wiki/Sistema_de_arquivos) e uma interface para o usuário, etc.), servindo como o intermediário entre o homem e a máquina. Ou seja, o SO faz a conexão entre o hardware e os demais software de um computador, interpreta os comandos do usuário, fornece a plataforma para que os programas sejam executados, organiza e gerencia os arquivos armazenados, como também controla a Entrada e Saída de dados dos dispositivos internos e periféricos.

Diagrama

Descrição gerada automaticamente

Em suma, as principais funções básicas de um sistema operacional são:

### Gerenciamento de Processos:

Um processo ou tarefa é uma porção de um programa (sequência de instruções a serem executadas pelo computador) em alguma fase de execução. Um programa pode consistir em várias tarefas, cada uma com funcionamento próprio ou como uma unidade, comunicando-se entre si periodicamente.

Tratando-se de ser um ambiente multitarefas, o **sistema operacional decide qual processo deve receber prioridade, ser processado primeiro e por quanto tempo**. Essa função é conhecida como agendamento de processos e é realizada pelo escalonador (Scheduler) da CPU (Unidade Central de Processamento).

Os processadores executam milhões de instruções por segundo. O SO é preparado para dar ao usuário a ilusão que o número de processos em execução no computador é maior que o número de processadores instalados, uma vez que cada processo é executado por um curto período e a alternância entre vários processos é tão rápida que o usuário pensa que sua execução é simultânea. Normalmente, são utilizados [algoritmos de escalonamento](https://pt.wikipedia.org/wiki/Escalonamento_de_processos#Algoritmos_de_escalonamento) para determinar qual processo será executado em determinado momento e por quanto tempo.

### Gerenciamento de Memória:

### Refere-se ao gerenciamento da memória primária ou principal. A memória primária é um longo conjunto de palavras ou bytes, sendo que cada um deles tem seu próprio endereço dentro da máquina.

### O sistema operacional tem acesso completo à memória do sistema e deve permitir que os processos dos usuários tenham acesso seguro à memória quando o requisitam. Por isso, é papel do SO garantir que cada programa tenha sua área de memória protegida de outros programas para que um não acesse nenhum endereço de memória que esteja em uso por outro.

Ademais, a memória primária providencia um armazenamento rápido que pode ser acessado diretamente pela CPU. Logo, para o sistema operacional conseguir executar um software qualquer, ele precisa acessar essa memória principal — e é isso que o gerenciamento de memória permite. Ou seja, essa funcionalidade **permite a execução dos softwares pelo computador**.

### Outra função semelhante é assegurar que os dados de cada usuário sejam gravados de forma confiável, considerando que existem vários programas, de vários usuários, gravando dados de forma concorrente. Neste caso, cabe ao SO garantir a integridade dos dados armazenados pelos vários programas

### Gerenciamento de Dispositivo:

O sistema operacional cria uma**comunicação direta por meio de drivers com os dispositivos conectados no hardware**. Um driver, portanto, é um software que viabiliza a execução de um dispositivo, que pode ser um pen drive, mouse, teclado ou até mesmo a placa de vídeo da máquina. Portanto o driver é uma espécie de tradutor, permitindo que a comunicação seja feita entre hardware e o software (sistema operacional).

### Entrada e Saída de Dados:

### Uma das funções principais de um sistema operacional é controlar todos os dispositivos de e/s de um computador, sejam internos (CPU, HD, placa mãe, pentes de memória, placa de vídeo etc.) e externos (monitor, mouse, teclado, caixa de som etc.). A maioria do hardware do dispositivo utiliza uma interface de baixo nível e complexa. O SO serve para fornecer uma interface de alto nível entre o hardware e o usuário.

### Segurança:

Um sistema operacional é capaz de**evitar acesso de agentes não autorizados**a arquivos ou configurações por meio de senhas e outros métodos de segurança.

### Controle da performance do sistema:

Sistemas operacionais também são responsáveis por **manter registros da performance do sistema em geral**, caso aconteça algum imprevisto e o serviço demore a receber uma resposta do sistema.

### Ajuda de detecção de erros:

**O sistema operacional produz mensagens de erro e relatórios sempre que um serviço apresenta um**[**bug**](https://blog.betrybe.com/tecnologia/o-que-e-bug/) **(falha no sistema)**, que ajudam no processo de correção do erro no código do sistema — ou [**debug**](https://blog.betrybe.com/tecnologia/debug/)**.**

### Administração de serviços:

O SO **mantém controle e administra o tempo e os recursos que cada serviço precisa para seu funcionamento**. Por exemplo, é dever do sistema operacional garantir que nenhum processo monopolize o uso da CPU, permitindo que haja uma alternância dos processos executados concorrentemente.

### Sistema de Arquivos:

Refere-se ao gerenciamento da memória secundária do computador. Enquanto a memória principal é volátil, ou seja, é mantido temporariamente e de forma não permanente, a memória secundária é do tipo não volátil, não podendo ser endereçada diretamente pela CPU, e é geralmente gravada em discos rígidos, mídias ópticas como CDs e DVDs, cartões de memória e pendrives (flash drives) e discos flexíveis (disquetes), entre outros.

Um sistema de arquivos **permite o armazenamento organizado de arquivos**, agregando características a cada arquivo como um nome, permissões de acesso, atributos especiais e um índice, que é uma lista de arquivos na partição que informa onde cada arquivo está localizado no disco. Assim, o sistema operacional é capaz de encontrar o arquivo em seu local de armazenamento rapidamente.

Os arquivos podem ser acessados em um dispositivo de armazenamento por meio de interface gráfica ou linha de comandos, fornecida pelo próprio SO.

**Conclusão**

**Referências**

* <https://www.tecmundo.com.br/sistema-operacional/2031-a-historia-dos-sistemas-operacionais-ilustracao-.htm>
* <file:///C:/Users/joaop/Downloads/a-historia-dos-sistemas-operacionais.pdf>
* <https://pt.wikibooks.org/wiki/Sistemas_operacionais/Hist%C3%B3ria#1964_-_Multics>
* <http://professor.pucgoias.edu.br/sitedocente/admin/arquivosUpload/17510/material/Historico_Evolucao_Sistemas_Operacionais.pdf>
* <http://ninjadolinux.com.br/a-historia-do-unix/#:~:text=O%20primeiro%20sistema%20UNIX%20foi,)%2C%20criado%20apenas%20por%20Ken>.
* <https://tecnoblog.net/responde/o-que-e-um-sistema-operacional/#:~:text=%C3%89%20um%20software%20ou%20conjunto,o%20computador%20e%20o%20usu%C3%A1rio>
* <https://www.rankmyapp.com/pt-br/conheca-os-principais-recursos-e-inovacoes-das-versoes-de-android/#:~:text=1.5%20Android%20Cupcake&text=Funcionando%20com%20base%20no%20Linux,com%20Gmail%2C%20entre%20outras%20funcionalidades>.
* <https://blog.bemmaisseguro.com/steve-jobs-e-bill-gates/>
* <https://www.rankmyapp.com/pt-br/conheca-os-principais-recursos-e-inovacoes-das-versoes-de-android/#:~:text=1.5%20Android%20Cupcake&text=Funcionando%20com%20base%20no%20Linux,com%20Gmail%2C%20entre%20outras%20funcionalidades>.
* <https://www.istoedinheiro.com.br/7-inovacoes-da-apple-que-foram-copiadas-pelas-fabricantes-de-android/>
* <https://www.oficinadanet.com.br/smartphones/33181-todos-os-iphones-ja-lancados>
* <https://velhobit.com.br/editorial/sistemas-operacionais-parte-1-historia.html>
* <https://olhardigital.com.br/2020/11/20/noticias/anos-de-evolucao-conheca-a-historia-do-windows/>
* <https://canaltech.com.br/windows/do-10-ao-11-veja-como-o-windows-evoluiu-a-cada-geracao-190109/>
* <https://blog.grancursosonline.com.br/o-que-e-um-sistema-operacional/>
* <https://blog.betrybe.com/tecnologia/sistema-operacional-tudo-sobre/>
* <https://sites.google.com/site/proffernandosiqueiraso/aulas/2-a-evolucao-dos-sistemas-operacionais>
* <https://techlib.wiki/definition/operating_system.html>
* <https://tecnoblog.net/responde/o-que-e-um-sistema-operacional/>
* <https://sites.google.com/site/proffernandosiqueiraso/aulas/4-estrutura-do-sistema-operacional>
* <https://dev.to/nfo94/entendendo-a-estrutura-de-um-sistema-operacional-i1n#:~:text=Um%20sistema%20operacional%20%C3%A9%20formado,elimina%C3%A7%C3%A3o%20de%20processos%20e%20threads>
* <https://www.tecmundo.com.br/aumentar-desempenho/11266-como-funciona-o-boot-de-um-computador.htm>
* <https://www.programaria.org/o-que-e-o-terminal-ou-venha-conhecer-tela-preta/>
* <https://pt.wikipedia.org/wiki/Interface_de_base_texto>
* <https://www.tecmundo.com.br/historia/9528-a-historia-da-interface-grafica.htm>
* <https://pt.vecteezy.com/arte-vetorial/4329343-voz-pesquisa-smartphone-interface-vetor-modelo-discurso-reconhecimento-movel-app-interface-azul-design-layout-tela-voz-acao-e-controle-ui-plano-para-aplicativo-telefone-display-com-microfone>
* <https://blog.4partner.com.br/licenca-oracle-com-vmware-oracle-vm-aws/definicao-visual-de-virtualizacao/>
* <https://www.devmedia.com.br/como-funcionam-os-dispositivos-de-entrada-e-saida/28275#:~:text=Uma%20das%20fun%C3%A7%C3%B5es%20principais%20de,emitir%20comandos%20para%20os%20dispositivos>.
* <https://rockcontent.com/br/blog/bash/>
* <http://www.bosontreinamentos.com.br/hardware/o-que-e-um-sistema-de-arquivos-file-system/#:~:text=Um%20Sistema%20de%20Arquivos%20%C3%A9,(disquetes)%2C%20entre%20outros>.
* <https://www.oficinadanet.com.br/artigo/851/o_que_e_um_sistema_operacional>
* <https://pt.wikipedia.org/wiki/Sistema_operativo#Gerenciamento_de_mem%C3%B3ria>
* <https://www.inf.ufsc.br/~j.barreto/cca/sisop/sisoperac.html>
* <https://docplayer.com.br/2567344-Arquitetura-de-sistemas-operacionais-machado-maia-arquitetura-de-sistemas.html>
* <https://www.psafe.com/blog/sistema-operacional-como-seu-pc-e-celular-funcionam/>
* <https://slideplayer.com.br/amp/6856346/>
* <https://www.techtudo.com.br/noticias/2012/02/o-que-e-bootloader.ghtml>
* <https://canaltech.com.br/produtos/drivers-entenda-o-que-sao-e-como-eles-funcionam-195604/>
* <https://blog.maxieduca.com.br/memoria-computador/>
* <https://todasasrespostas.pt/o-que-e-e-como-funciona-o-sistema-operacional>
* <https://www.cursosdeinformaticabasica.com.br/o-que-e-sistema-operacional/>
* <https://pt.wikibooks.org/wiki/Arquitetura_entre_Sistemas_Operativos/Estrutura_dos_Sistemas_Operacionais>