



MINISTÉRIO DA EDUCAÇÃO
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE SÃO PAULO
UNIDADE DE ENSINO SALTO

**Curso: TÉCNICO EM INFORMÁTICA
com Habilitação em Programação e Desenvolvimento de
Sistemas.**

INTRODUÇÃO AOS SISTEMAS OPERACIONAIS ISO

**NOTA DE AULA
(TEORIA)**

SALTO/SP

Nota de aula

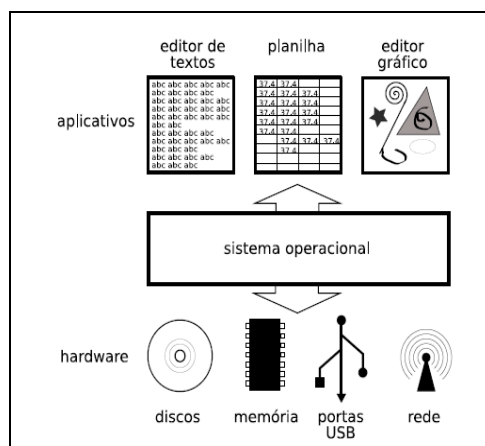
1. Caracterização de sistemas operacionais

DEFINIÇÃO

Principal software instalado no computador e é responsável pela interação entre o hardware e os softwares, também responsável pela manipulação de arquivos, sendo as suas principais operações de arquivo o armazenamento, a criação, a cópia, a remoção e a atribuição.

O sistema operacional é uma camada de software que opera entre o hardware e os programas aplicativos voltados ao usuário final. O sistema operacional é uma estrutura de software ampla, muitas vezes complexa, que incorpora aspectos de baixo nível (como drivers de dispositivos e gerência de memória física) e de alto nível (como programas utilitários e a própria interface gráfica).

A figura abaixo ilustra a arquitetura geral de um sistema de computação típico. Nela, podemos observar elementos de hardware, o sistema operacional e alguns programas aplicativos.



1.1 Histórico

A evolução dos sistemas operacionais está, em grande parte, relacionada ao desenvolvimento de equipamentos cada vez mais velozes, compactos e de custos baixos e a necessidade de aproveitamento e controle desses recursos.

Neste histórico dividimos essa evolução em fases, onde destacamos, em cada uma, suas principais características de hardware, software, interação com o sistema e aspectos de conectividade.

1.1.1 Primeira Fase (1945-1955) / Primeira Geração de Computadores

No início da Segunda Guerra Mundial, surgiram os primeiros computadores digitais, formados por milhares de válvulas, que ocupavam áreas enormes, sendo de funcionamento lento e duvidoso.

O ENIAC (Electronic Numerical Integrator and Computer) foi o primeiro computador digital de propósito geral. Criado para a realização de cálculos balísticos sua estrutura possuía 18 mil válvulas, 10 mil capacitores, 70 mil resistores e pesava 30 toneladas. Quando em operação, consumia cerca de 140 KW/h e era capaz de realizar cinco mil adições por segundo.

Para trabalhar nessas máquinas, era necessário conhecer profundamente o funcionamento do hardware, pois, a programação era feita em painéis, através de fios,

utilizando linguagem de máquina. Nessa fase, ainda não existia o conceito de sistema operacional.

Outros computadores foram construídos nessa mesma época, como o EDVAC (Electronic Discrete Variable Automatic Computer) e o IAS (Princeton Institute for Advanced Studies), mas eram utilizados, praticamente, apenas nas universidades e nos órgãos militares.

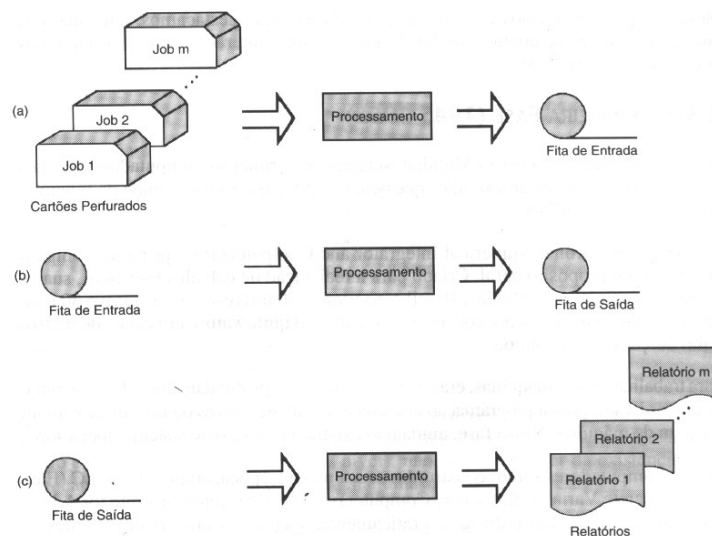
Com o desenvolvimento da indústria de computadores, muitas empresas foram fundadas ou investiram no setor, como a Sperry e a IBM, o que levou a criação dos primeiros computadores para aplicações comerciais. A primeira máquina fabricada com esse propósito e bem-sucedida foi o UNIVAC I (Universal Automatic Computer), criado especialmente para o censo americano de 1950.

1.1.2 Segunda Fase (1956-1965) / Segunda Geração de Computadores

A criação do transistor e das memórias magnéticas contribuiu para o enorme avanço dos computadores da época. O transistor permitiu o aumento da velocidade e da confiabilidade do processamento, e as memórias magnéticas permitiram o acesso mais rápido aos dados, maior capacidade de armazenamento e computadores menores.

Com o surgimento das primeiras linguagens de programação, como Assembly e Fortran, os programas deixaram de ser feitos diretamente no hardware, o que facilitou enormemente o processo de desenvolvimento de programas.

Já não era mais possível conviver com tantos procedimentos manuais como os anteriores, que não permitiam o uso eficiente do computador e de seus recursos. Os primeiros sistemas operacionais surgiram, justamente, para tentar automatizar as tarefas manuais



Processamento batch

Inicialmente, os programas passaram a ser perfurados em cartões que submetidos a uma leitora, eram gravados em uma fita de entrada. A fita, então, era lida pelo computador que executava um programa de cada vez, gravando o resultado do processamento em uma fita de saída. Ao terminar de todos os programas a fita de saída era lida e impressa. A esse tipo de processamento, onde um lote (batch) de programas era submetido ao computador deu-se o nome de processamento batch.

Pode não parecer um avanço, mas anteriormente os programas eram submetidos pelo operador, um a um, fazendo com que o processador ficasse ocioso entre a execução dos programas. Com o processamento batch, um grupo de programas era submetido de uma só vez, o que diminuía o tempo existente entre a execução dos programas, permitindo, assim, melhor uso do processador.

Os sistemas operacionais passaram a ter seu próprio conjunto de rotinas para operações de entrada/saída (Input/Output Control System—IOCS), que veio facilitar bastante o processo de programação. O IOCS eliminou a necessidade de os programadores desenvolverem suas próprias rotinas de leitura/gravação específicas para cada dispositivo periférico. Essa facilidade de comunicação criou o conceito de independência de dispositivos.

Importantes avanços em nível de hardware foram implementados no final dessa fase, principalmente na linha 7094 da IBM. Entre eles destacamos o conceito de canal, que veio permitir a transferência de dados entre dispositivos de entrada/saída e memória principal de forma independente da UCP. Ainda nessa fase destacamos os sistemas FMS (Fortran Monitor System) e IBSYS.

1.1.3 Terceira Fase (1966-1980) / Terceira Geração de Computadores

Através dos circuitos integrados (CIs) e posteriormente dos microprocessadores foi possível viabilizar e difundir o uso de sistemas computacionais por empresas, devido a diminuição de seus custos de aquisição. Além disso houve grande aumento do poder de processamento e diminuição no tamanho dos equipamentos.

Com base nessa nova tecnologia a IBM lançou em 1964 a Série 360. Esse lançamento causou uma revolução na indústria de informática, pois, introduzia uma linha (família) de computadores pequena, poderosa e principalmente compatível. Isso permitiu que uma empresa adquirisse um modelo mais simples e barato e conforme suas necessidades, mudasse para modelos com mais recursos sem comprometer suas aplicações já existentes. Para essa série foi desenvolvido o sistema operacional OS/360, que tentava atender todos os tipos de aplicações e periféricos. Apesar de todos os problemas desse equipamento e de seu tamanho físico a Série 360 introduziu novas técnicas utilizadas até hoje.

Na mesma época, a DEC lançou a linha PDP-8, também revolucionária, pois, apresentava uma linha de computadores de porte pequeno e baixo custo, se comparada aos mainframes até então comercializados, criando um novo mercado, o de minicomputadores.

A evolução dos processadores de entrada/saída permitiu que enquanto um programa esperasse por uma operação de leitura/gravação o processador executasse um outro programa. Para tal a memória foi dividida em partições, onde cada programa esperava sua vez para ser processado. A essa técnica de compartilhamento da memória principal e processador deu-se o nome de multiprogramação.

Com a substituição das fitas por discos no processo de submissão dos programas, o processamento batch tornou-se mais eficiente, pois, permitia a alteração na ordem de execução das tarefas, até então puramente seqüencial. A essa técnica de submissão de programas chamou-se spooling que mais tarde, também viria a ser utilizada no processo de impressão.

Os sistemas operacionais, mesmo implementando o processamento batch e a multiprogramação ainda estavam limitados aos processamentos que não exigiam comunicação com o usuário. Para permitir a interação rápida entre o usuário e o computador, foram adicionados terminais de vídeo e teclado (interação on-line).

A multiprogramação evoluiu preocupada em oferecer aos usuários tempos de respostas razoáveis e uma interface cada vez mais amigável. Para tal, cada programa na memória utilizaria o processador em pequenos intervalos de tempo. A esse sistema de divisão de tempo do processador chamou-se time-sharing (tempo compartilhado).

Outro fato importante nessa fase foi o surgimento do sistema operacional Unix (1969). Concebido inicialmente em um minicomputador PDP-7, baseado no sistema MULTICS (Multiplexed Information and Computing Service), o Unix foi depois reescrito em uma linguagem de alto nível (linguagem C), tornando-se conhecido por sua portabilidade.

No final dessa fase, com a evolução dos microprocessadores, surgiram os primeiros microcomputadores, muito mais baratos que qualquer um dos computadores até então

comercializados. Entre eles, destacamos os micros de 8 bits da Apple e o sistema operacional CP/M (Control Program Monitor).

1.1.4 Quarta Fase (1981-1990) / Quarta Geração de Computadores

A integração em larga escala (Large Scale Integration-LSI) e a integração em muito larga escala (Very Large Scale Integration-VLSI) levaram adiante o projeto de miniaturização e barateamento dos equipamentos. Os mini e superminicomputadores se firmaram no mercado e os microcomputadores ganharam um grande impulso.

Nesse quadro surgiram os microcomputadores PC (Personal Computer) de 16 bits da IBM e o sistema operacional DOS (Disk Operation System), criando a filosofia dos computadores pessoais. Na área dos minis e superminicomputadores ganharam impulso os sistemas multiusuário, com destaque para os sistemas compatíveis com o Unix (Unix-like) e o VMS (Virtual Memory System) da DEC. Surgem as estações de trabalho (workstations) que apesar de monousuárias permitem que se executem diversas tarefas concorrentemente, criando o conceito de multitarefa.

No final dos anos 80, os computadores tiveram um grande avanço, decorrente de aplicações que exigiam um enorme volume de cálculos. Para acelerar o processamento, foram adicionados outros processadores, exigindo dos sistemas operacionais novos mecanismos de controle e sincronismo. Com o multiprocessamento, foi possível a execução de mais de um programa simultaneamente ou até de um mesmo programa por mais de um processador. Além de equipamentos com múltiplos processadores foram introduzidos processadores vetoriais e técnicas de paralelismo em diferentes níveis, fazendo com que os computadores se tornassem ainda mais poderosos.

As redes distribuídas (Wide Area Network- WANs) se difundiram por todo o mundo, permitindo o acesso a outros sistemas de computação independentemente de estado, de país e, até mesmo de fabricante. Nesse contexto são desenvolvidos inúmeros protocolos de rede, alguns proprietários, como o DECnet da DEC e o SNA (System Network Architecture) da IBM e outros de domínio público, como o TCP/IP e o CCITT X.25. Surgem as primeiras redes locais (Local Area Network—LANs) interligando pequenas áreas. Os softwares de rede passaram a estar intimamente relacionados ao sistema operacional e surgem os sistemas operacionais de rede.

1.1.5 Quinta Fase (1991-)

Grandes avanços em termos de hardware, software e telecomunicações podem ser esperados até o final deste século. Essas mudanças são consequência da evolução das aplicações que necessitam cada vez mais de capacidade de processamento e armazenamento de dados. Sistemas especialistas, sistemas multimídia, banco de dados distribuídos, inteligência artificial e redes neurais são apenas alguns exemplos da necessidade cada vez maior.

A evolução da microeletrônica permite o desenvolvimento de processadores e memórias cada vez mais velozes e baratos além de dispositivos menores, mais rápidos e com maior capacidade de armazenamento. Os componentes baseados em tecnologia VLSI (Very Large Scale Integration) evoluíram rapidamente para o ULSI (Ultra Large Scale Integration).

Os computadores da próxima geração têm de ser muito mais eficientes que os atuais, para atender o volume cada vez maior de processamento. Para isso está ocorrendo uma mudança radical na filosofia de projeto de computadores. Arquiteturas paralelas baseadas em organizações de multiprocessadores não convencionais já se encontram em desenvolvimento em várias universidades e centros de pesquisa do mundo.

A evolução do hardware encadeará modificações profundas nas disciplinas de programação para fazer melhor uso das arquiteturas paralelas. Assim novas linguagens e metodologias de programação concorrentes estão sendo desenvolvidas, em particular, fazendo uso extensivo de inteligência artificial e CAD (Computer-Aided Design).

O conceito de processamento distribuído será explorado nos sistemas operacionais de forma que suas funções estejam espalhadas por vários processadores através de redes de computadores. Isso só será possível devido a redução dos custos de comunicação e ao aumento na taxa de transmissão de dados.

A arquitetura cliente-servidor aplicada basicamente a redes locais passe a ser oferecida em redes distribuídas permitindo que qualquer pessoa tenha acesso a todo tipo de informação independentemente de onde esteja armazenada. Problemas de segurança, gerência e desempenho tornam-se fatores importantes relacionados ao sistema operacional e a rede.

A década de 90 foi definitiva para a consolidação dos sistemas operacionais baseados em interfaces gráficas. Novas interfaces homem-máquina estão sendo utilizadas, como linguagens naturais, sons e imagens, fazendo essa comunicação mais inteligente, simples e eficiente.

Os conceitos e implementações só vistos em sistemas considerados de grande porte estão sendo introduzidos na maioria dos sistemas desktop, como na família Windows da Microsoft, no Unix e no OS/2 da IBM.

Fase	Primeira (1945-1955)	Segunda (1956-1965)	Terceira (1966-1980)	Quarta (1981-1990)	Quinta (1991-)
Computadores	ENIAC EDVAC UNIVAK	NCR IMB 7094 CDC-6600	IBM 360, 370 PDP-11 Cray 1 Cyber-205	Cray XMP IBM 308 VAX-11 IBM-PC	IBM 3090 Alpha AXP Pentium Sun SPARC
Hardware	Válvulas Tambor Magnético Tubos de raios catódicos	Transistor Memória Magnética	Circuito Integrado Disco Magnético Minicomputador Microprocessador	LSI ou VLSI Disco óptico Microcomputador	Ultra-LSI Arquiteturas Paralelas Circuito Integrado 3-D
Software	Linguagem de Máquina Linguagem assembly	Linguagem de Alto Nível Processamento Batch	Linguagem Estruturadas Multiprogramação Time-Sharing Computação Gráfica	Multiprocessamento Sistemas Especialistas Linguagens orientadas a objetos	Processamento Distribuído Linguagens concorrentes Programação funcional Linguagens naturais
Telecomunicações	Telefone Teletipo	Transmissão Digital	Comunicação via satélite Microondas Redes distribuídas(WAN) Fibra óptica	Redes Locais (LAN) Internet	Redes Locais estendidas(ELAN) Redes sem fio Modelo cliente-servidor
Desempenho	10 ips	200.000 ips	5 Mips	30 Mips	1 Gflops 1 Tflops

1.2 Abstração de recursos

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. Por exemplo, a seqüência a seguir apresenta os principais passos envolvidos na abertura de um arquivo (operação open) em um leitor de disquete:

- 1° . verificar se os parâmetros informados estão corretos (nome do arquivo, identificador do leitor de disquete, buffer de leitura, etc);
- 2° . verificar se o leitor de disquetes está disponível;
- 3° . verificar se o leitor contém um disquete;
- 4° . ligar o motor do leitor e aguardar atingir a velocidade de rotação correta;
- 5° . posicionar a cabeça de leitura sobre a trilha onde está a tabela de diretório;
- 6° . ler a tabela de diretório e localizar endereço do arquivo ou subdiretório desejado;
- 7° . mover a cabeça de leitura para a posição do bloco inicial do arquivo;
- 8° . ler o bloco do arquivo e depositá-lo em um buffer de memória.

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

Prover interfaces de acesso aos dispositivos mais simples de usar que as interface de baixo nível para simplificar a construção de programas aplicativos. Por exemplo: para ler dados de um disco rígido, uma aplicação usa um conceito chamado *arquivo*, que implementa uma visão abstrata do disco rígido, acessível através de operações como open e read. Caso tivesse de acessar o disco diretamente, teria de manipular portas de entrada/saída e registradores com comandos para o controlador de disco (sem falar na dificuldade de localizar os dados desejados dentro do disco).

Ao tornar os aplicativos independentes do hardware definindo uma interface abstrata de acesso a um dispositivo de hardware o sistema operacional desacopla o hardware dos aplicativos e permite que ambos evoluam de forma mais autônoma. Por exemplo, o código de um editor de textos não deve ser dependente da tecnologia de discos rígidos utilizada no sistema.

Definir interfaces de acesso homogêneas para dispositivos com tecnologias distintas. Através de suas abstrações, o sistema operacional permite aos aplicativos usar a mesma interface para dispositivos diversos. Por exemplo, um aplicativo acessa dados em disco através de arquivos e diretórios sem precisar se preocupar com a estrutura real de armazenamento dos dados que podem estar em um disquete, em um disco IDE, em uma máquina fotográfica digital conectada à porta USB, em um CD ou mesmo num disco remoto compartilhado através da rede.

1.3 Gerência de recursos

Os programas aplicativos usam o hardware para atingir seus objetivos: ler e armazenar dados, editar e imprimir documentos, navegar na Internet, tocar música e etc. Em um sistema com várias atividades simultâneas podem surgir conflitos no uso do hardware quando dois ou mais aplicativos precisam dos mesmos recursos. Cabe ao sistema operacional definir políticas para gerenciar o uso dos recursos de hardware pelos aplicativos e resolver eventuais disputas e conflitos. Vejamos uma situação onde a gerência de recursos do hardware se faz necessária:

Cada computador possui normalmente um só processador. O uso desse processador deve ser distribuído entre os aplicativos presentes no sistema de forma que cada um deles possa executar na velocidade adequada para cumprir suas funções sem prejudicar os outros. O mesmo ocorre com a memória RAM, que deve ser distribuída de forma justa entre as aplicações.

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, independentes das características e detalhes de baixo nível e quais os conflitos no uso do hardware são minimizados.

2. Tipos de sistemas operacionais

Os sistemas operacionais podem ser classificados segundo diversos parâmetros e perspectivas, como tamanho, velocidade, suporte a recursos específicos, acesso à rede, etc. A seguir são apresentados alguns tipos de sistemas operacionais usuais (muitos sistemas operacionais se encaixam bem em mais de uma das categorias apresentadas).

2.1 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 um programa após o outro, processando-os em seqüência, o que permitia um alto grau de utilização do sistema. Ainda hoje o termo “em lote” é usado

para designar um conjunto de comandos que deve ser executado em seqüência, sem interferência do usuário.

Exemplos desses sistemas incluem o OS/360 e VMS, entre outros.

2.2 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. Ele também deve disponibilizar seus recursos locais aos demais computadores, de forma controlada.

A maioria dos sistemas operacionais atuais oferece esse tipo de funcionalidade.

2.3 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. Os sistemas operacionais distribuídos já existem há tempos (Amoeba e Clouds, por exemplo), mas ainda não são uma realidade de mercado.

2.4 Multi-usuário

Um sistema operacional multi-usuário 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. Essa funcionalidade é fundamental para a segurança dos sistemas operacionais de rede e distribuídos. Grande parte dos sistemas atuais são multi-usuários.

2.5 Desktop

Um sistema operacional “de mesa” é voltado ao atendimento do usuário doméstico e corporativo para a realização de atividades corriqueiras, como edição de textos e gráficos, navegação na Internet e reprodução de mídias simples. Sua principais características são a interface gráfica, o suporte à interatividade e a operação em rede. Exemplos de sistemas desktop são o Windows XP, MacOS X e Linux.

2.6 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. Normalmente um sistema operacional servidor também tem suporte a rede e multi-usuários.

2.7 Embutido

Um sistema operacional é dito embutido (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, controladores industriais e automotivos, equipamentos eletrônicos de uso doméstico (leitores de DVD, TVs, fornos micro-ondas, centrais de alarme, etc.). Muitas vezes um sistema operacional embutido

se apresenta na forma de uma biblioteca a ser ligada ao programa da aplicação (que é fixa). Exemplos de sistemas operacionais embutidos são o μ C/OS, Xylinx, LynxOS e VxWorks.

2.8 Tempo real

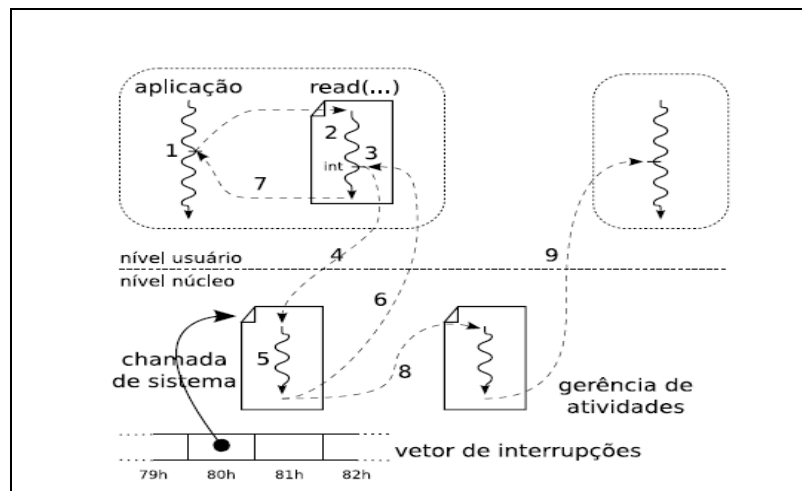
Ao contrário da concepção usual, um sistema operacional de tempo real não precisa ser necessariamente ultra-rápido; sua característica essencial é ter um comportamento temporal previsível (ou seja, seu tempo de resposta deve ser conhecido no melhor e pior caso de operação). A estrutura interna de um sistema operacional de tempo real deve ser construída de forma a minimizar esperas e latências imprevisíveis, como tempos de acesso a disco e sincronizações excessivas.

Existem duas classificações de sistemas de tempo real: soft real-time systems, nos quais a perda de prazos implica na degradação do serviço prestado. Um exemplo seria o suporte à gravação de CDs ou à reprodução de músicas. Caso o sistema se atrase, pode ocorrer a perda da mídia em gravação ou falhas na música que está sendo tocada. Por outro lado, nos hard real-time systems a perda de prazos pelo sistema pode perturbar o objeto controlado, com graves conseqüências humanas, econômicas ou ambientais. Exemplos desse tipo de sistema seriam o controle de funcionamento de uma turbina de avião a jato ou de uma caldeira industrial.

Exemplos de sistemas de tempo real incluem QNX, RT-Linux e VxWorks. Muitos sistemas embutidos têm características de tempo real, e vice-versa.

3 Arquiteturas de Sistemas Operacionais

Embora a definição de níveis de privilégio imponha uma estruturação mínima a um sistema operacional, as múltiplas partes que compõem o sistema podem ser organizadas de diversas formas, separando suas funcionalidades e modularizando seu projeto. Nesta seção serão apresentadas as arquiteturas mais populares para a organização de sistemas operacionais.

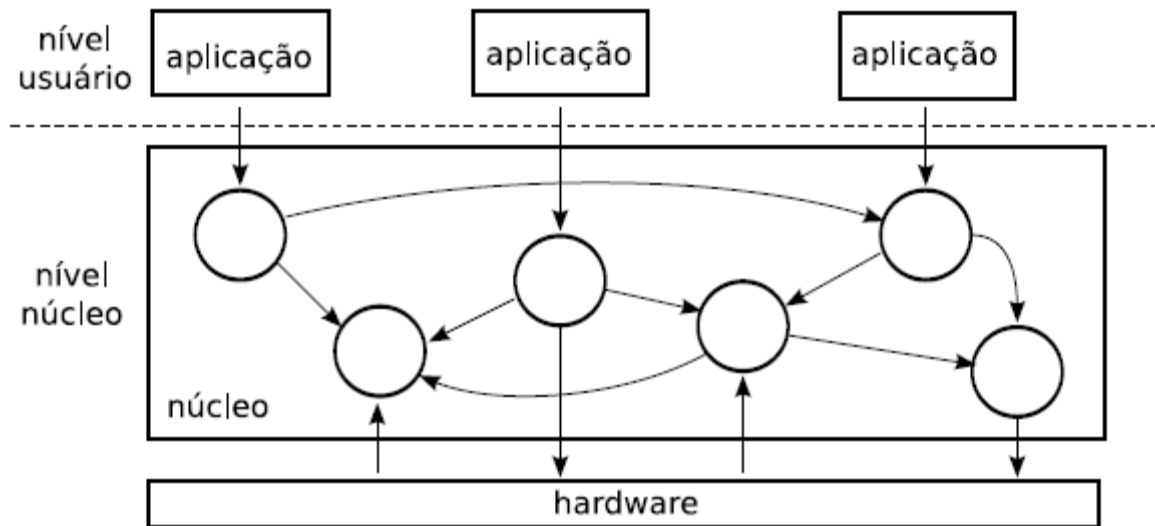


Roteiro típico de uma chamada de sistema

3.1 Sistemas monolíticos

Em um sistema monolítico, todos os componentes do núcleo operam em modo núcleo e se inter-relacionam conforme suas necessidades, sem restrições de acesso entre si, pois o

código no nível núcleo tem acesso pleno a todos os recursos e áreas de memória.



Uma arquitetura monolítica

A grande vantagem dessa arquitetura é seu desempenho: qualquer componente do núcleo pode acessar os demais componentes, toda a memória ou mesmo dispositivos periféricos diretamente, pois não há barreiras impedindo esse acesso. A interação direta entre componentes também leva a sistemas mais compactos.

Todavia, a arquitetura monolítica pode pagar um preço elevado por seu desempenho: a robustez e a facilidade de desenvolvimento. Caso um componente do núcleo perca o controle devido a algum erro, esse problema pode se alastrar rapidamente por todo o núcleo, levando o sistema ao colapso (travamento, reinicialização ou funcionamento errado). Além disso, a manutenção e evolução do núcleo se tornam mais complexas, porque as dependências e pontos de interação entre os componentes podem não ser evidentes: pequenas alterações na estrutura de dados de um componente podem ter um impacto inesperado em outros componentes, caso estes acessem aquela estrutura diretamente.

A arquitetura monolítica foi a primeira forma de organizar os sistemas operacionais; sistemas UNIX antigos e oMS-DOS seguiam esse modelo. Atualmente, apenas sistemas operacionais embutidos usam essa arquitetura, devido às limitações do hardware sobre o qual executam.

O núcleo do Linux nasceu monolítico, mas vem sendo paulatinamente estruturado e modularizado desde a versão 2.0 (embora boa parte de seu código ainda permaneça no nível de núcleo).

3.2 Sistemas em camadas

Uma forma mais elegante de estruturar um sistema operacional faz uso da noção de camadas: a camada mais baixa realiza a interface como hardware, enquanto as camadas intermediárias provêm níveis de abstração e gerência cada vez mais sofisticados. Por fim, a camada superior define a interface do núcleo para as aplicações (as chamadas de sistema). Essa abordagem de estruturação de software fez muito sucesso no domínio das redes de computadores, através do modelo de referência OSI (Open Systems Interconnection), e também seria de se esperar sua adoção no domínio dos sistemas operacionais. No entanto, alguns inconvenientes limitam sua aceitação nesse contexto:

- O empilhamento de várias camadas de software faz com que cada pedido de uma aplicação demore mais tempo para chegar até o dispositivo periférico ou recurso a ser acessado, prejudicando o desempenho do sistema.

- Não é óbvio como dividir as funcionalidades de um núcleo de sistema operacional em camadas horizontais de abstração crescente, pois essas funcionalidades são interdependentes, embora tratem muitas vezes de recursos distintos.

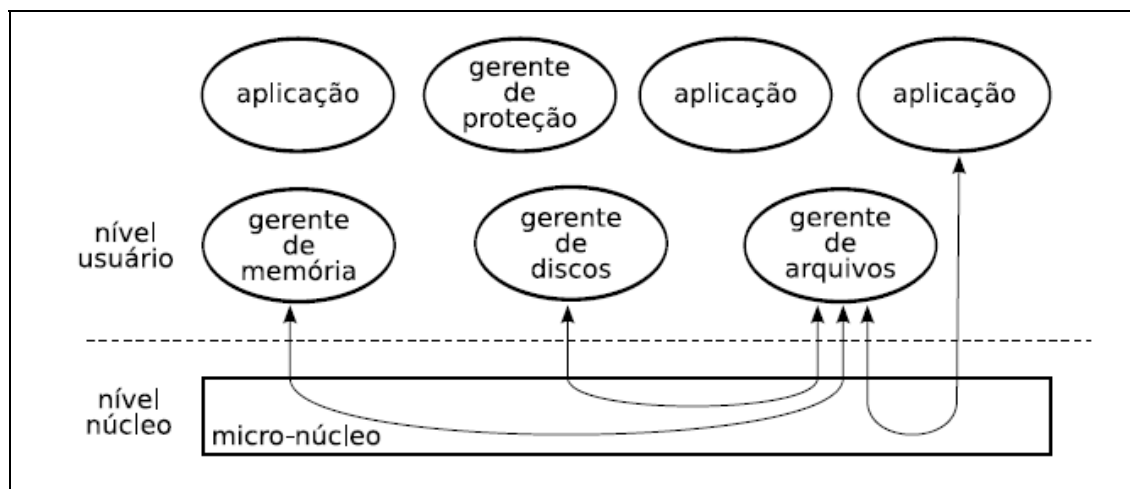
Em decorrência desses inconvenientes, a estruturação em camadas é apenas parcialmente adotada hoje em dia. Muitos sistemas implementam uma camada inferior de abstração do hardware para interagir com os dispositivos (a camada HAL – Hardware Abstraction Layer, implementada no Windows NT e seus sucessores), e também organizam em camadas alguns sub-sistemas como a gerência de arquivos e o suporte de rede (segundo o modelo OSI). Como exemplos de sistemas fortemente estruturados em camadas podem ser citados o IBM OS/2 e o MULTICS.

3.3 Sistemas micro-núcleo

Uma outra possibilidade de estruturação consiste em retirar do núcleo todo o código de alto nível (normalmente associado às políticas de gerência de recursos), deixando no núcleo somente o código de baixo nível necessário para interagir como hardware e criar as abstrações fundamentais (como a noção de atividade). Por exemplo, usando essa abordagem o código de acesso aos blocos de um disco rígido seria mantido no núcleo, enquanto as abstrações de arquivo e diretório seriam criadas e mantidas por um código fora do núcleo, executando da mesma forma que uma aplicação do usuário.

Por fazer os núcleos de sistema ficarem menores, essa abordagem foi denominada micro-núcleo (ou μ -kernel). Um micro-núcleo normalmente implementa somente a noção de atividade, de espaços de memória protegidos e de comunicação entre atividades.

Todos os aspectos de alto nível, como políticas de uso do processador e da memória, o sistema de arquivos e o controle de acesso aos recursos são implementados fora do núcleo, em processos que se comunicam usando as primitivas do núcleo.



Visão geral de uma arquitetura micro-núcleo

Em um sistema micro-núcleo, as interações entre componentes e aplicações são feitas através de trocas de mensagens. Assim, se uma aplicação deseja abrir um arquivo no disco rígido, envia uma mensagem para o gerente de arquivos que, por sua vez, se comunica com o gerente de dispositivos para obter os blocos de dados relativos ao arquivo desejado. Os processos não podem se comunicar diretamente, devido às restrições impostas pelos mecanismos de proteção do hardware. Por isso, todas as mensagens são transmitidas através de serviços do micro-núcleo. Como os processos têm de solicitar “serviços” uns dos outros, para poder realizar suas tarefas, essa abordagem também foi denominada cliente-servidor.

O micro-núcleos foram muito investigados durante os anos 80. Dois exemplos clássicos dessa abordagem são os sistemas Mach e Chorus. As principais vantagens dos sistemas micro-núcleo são sua robustez e flexibilidade: caso um sub-sistema tenha problemas,

os mecanismos de proteção de memória e níveis de privilégio irão confiná-lo, impedindo que a instabilidade se alastre ao restante do sistema. Além disso, é possível customizar o sistema operacional, iniciando somente os componentes necessários ou escolhendo os componentes mais adequados às aplicações que serão executadas.

Vários sistemas operacionais atuais adotam parcialmente essa estruturação; por exemplo, o Maços X da Apple tem suas raízes no sistema Mach, ocorrendo o mesmo com o Digital UNIX. Todavia, o custo associado às trocas de mensagens entre componentes pode ser bastante elevado, o que prejudica seu desempenho e diminui a aceitação desta abordagem. O QNX é um dos poucos exemplos de micro-núcleo amplamente utilizado, sobretudo em sistemas embutidos e de tempo-real.

3.4 Máquinas virtuais

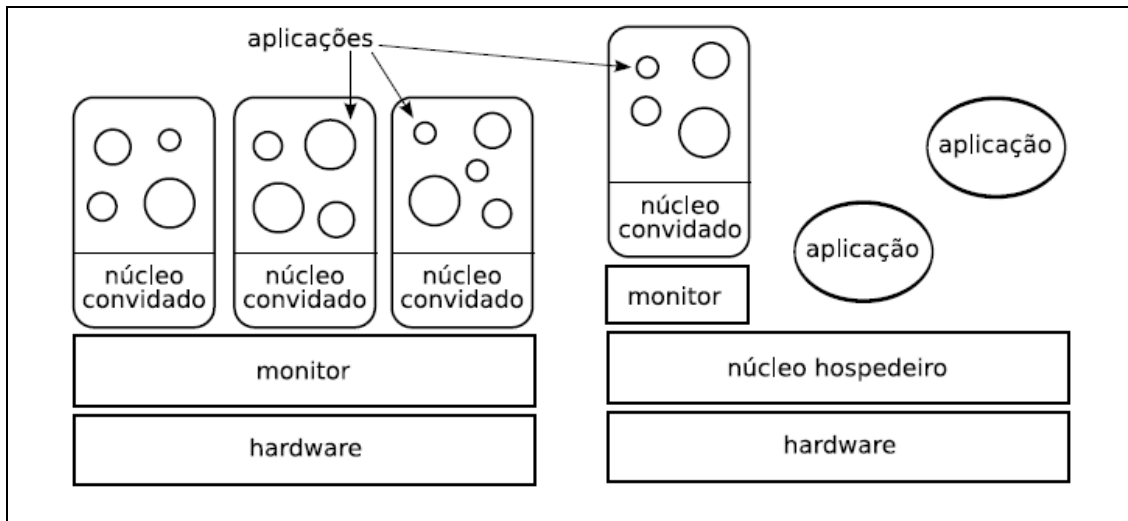
Em um computador real uma camada de software de baixo nível (por exemplo, a BIOS dos sistemas PC) fornece acesso aos vários recursos do hardware para o sistema operacional que os disponibiliza de forma abstrata às aplicações. Quando o sistema operacional acessa os dispositivos de hardware ele faz uso dos drivers respectivos que interagem diretamente com a memória e os dispositivos da máquina.

Por outro lado um emulador implementa todas as instruções realizadas pela máquina real em um ambiente abstrato, possibilitando executar um aplicativo de uma plataforma em outra, por exemplo, um aplicativo do Windows executando no Linux. Infelizmente, um emulador perde muito em eficiência ao traduzir as instruções de uma máquina real.

Além disso emuladores são bastante complexos, pois, geralmente necessitam simular a quase totalidade das instruções do processador e demais características do hardware que os circundam.

Uma máquina virtual é definida como “uma duplicata eficiente e isolada de uma máquina real”. A funcionalidade e o nível de abstração de uma máquina virtual se encontra entre uma máquina real e um emulador, na medida em que abstrai somente os recursos de hardware e de controle usados pelas aplicações. Uma máquina virtual é um ambiente criado por um monitor de máquinas virtuais, também denominado “sistema operacional para sistemas operacionais”. O monitor pode criar uma ou mais máquinas virtuais sobre uma única máquina real. Enquanto um emulador fornece uma camada de abstração completa entre o sistema em execução e o hardware, um monitor abstrai o hardware subjacente e controla uma ou mais máquinas virtuais. Cada máquina virtual fornece facilidades para uma aplicação ou um “sistema convidado” que acredita estar executando sobre um ambiente normal com acesso físico ao hardware.

Existem basicamente duas abordagens para a construção de sistemas de máquinas virtuais: o tipo I, onde o monitor de máquinas virtuais é implementado entre o hardware e os sistemas convidados, e o tipo II, onde o monitor é implementado como um processo de um sistema operacional real subjacente, denominado sistema anfitrião ou sistema hospedeiro. Ambas as abordagens estão ilustradas na figura abaixo, Como exemplos de sistemas de tipo I podem ser citados o VM Ware ESX Server e o Xen; para o tipo II podem ser citados o VM Ware Workstation, o MS Virtual PC e o User-Mode Linux.



Máquinas virtuais de tipo I (esquerda) e de tipo II (direita)

Como o sistema operacional convidado e o ambiente de execução na máquina virtual são idênticos ao da máquina real, é possível usar os softwares já construídos para a máquina real dentro das máquinas virtuais. Essa transparência evita ter de construir novas aplicações ou adaptar as já existentes.

As máquinas virtuais têm recebido bastante destaque nos últimos anos, sobretudo devido ao grande sucesso de linguagens independentes de plataforma como Java. Todavia, a máquina virtual Java (JVM – Java Virtual Machine) deve ser considerada basicamente um emulador, na medida em que cria um ambiente de execução abstrato para aplicações Java. Programas em linguagem Java são compilados em bytecodes, que são basicamente código de máquina para um processador abstrato. Assim, a JVM somente executa aplicações construídas especificamente para essa máquina hipotética. Outros emuladores populares são o QEmu e o Bochs.

Algumas vantagens são apresentadas para a utilização de máquinas virtuais em sistemas de computação:

- Aperfeiçoamento e testes de novos sistemas operacionais;
- Ensino prático de sistemas operacionais e programação de baixo nível;
- Executar diferentes sistemas operacionais sobre o mesmo hardware, simultaneamente;
 - Simular configurações e situações diferentes do mundo real, como por exemplo, mais memória disponível, outros dispositivos de E/S;
 - Simular alterações e falhas no hardware para testes ou reconfiguração de um sistema operacional, provendo confiabilidade e escalabilidade para as aplicações;
 - Garantir a portabilidade das aplicações legadas (que executariam sobre uma VM simulando o sistema operacional original);
 - Desenvolvimento de novas aplicações para diversas plataformas, garantindo a portabilidade destas aplicações;
 - Diminuir custos com hardware.

A principal desvantagem do uso de máquinas virtuais é o custo adicional de execução dos processos na máquina virtual em comparação com a máquina real. Esse custo é muito variável, podendo passar de 50% em plataformas sem suporte de hardware à virtualização, como os PCs de plataforma Intel mais antigos. Todavia, pesquisas recentes têm obtido a redução desse custo a patamares abaixo de 20%, graças sobretudo a ajustes no código do sistema hospedeiro. Esse problema não existe em ambientes cujo hardware oferece suporte à virtualização, como é o caso dos mainframes e dos processadores Intel/AMD mais recentes.

4. Gerenciamento de Memória

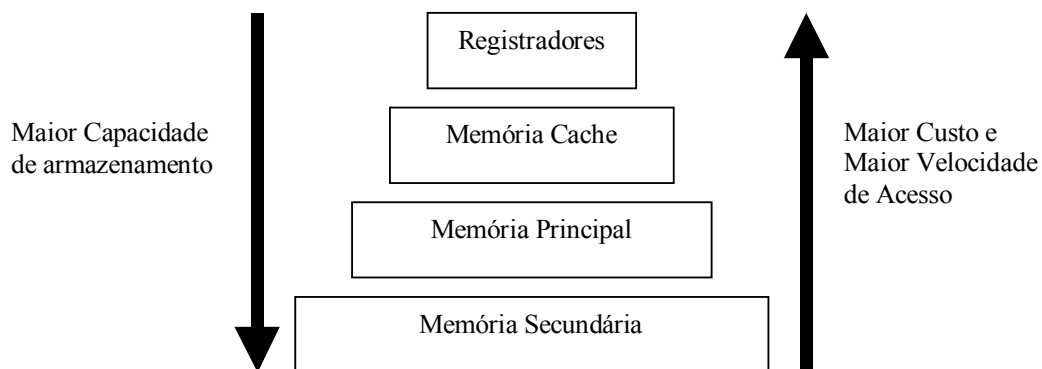
Tem como objetivo fornecer a cada aplicação uma área de memória própria, independente e isolada das demais aplicações e inclusive do núcleo do sistema. O isolamento das áreas de memória das aplicações melhora a estabilidade e segurança do sistema como um todo, pois impede aplicações com erros (ou aplicações maliciosas) de interferir no funcionamento das demais aplicações. Além disso, caso a memória RAM existente seja insuficiente para as aplicações, o sistema operacional pode aumentá-la de forma transparente às aplicações, usando o espaço disponível em um meio de armazenamento secundário (como um disco rígido). Uma importante abstração construída pela gerência de memória é a noção de memória virtual, que desvincula os endereços de memória vistos por cada aplicação dos endereços acessados pelo processador na memória RAM. Com isso, uma aplicação pode ser carregada em qualquer posição livre da memória, sem que seu programador tenha de se preocupar com os endereços de memória onde ela irá executar.

4.1 Volatilidade

A memória pode ser classificada em função de sua volatilidade que é a capacidade de a memória preservar o seu conteúdo mesmo sem uma fonte de alimentação estar ativa. As memórias chamadas voláteis se caracterizam por poderem ser lidas ou gravadas, como o tipo RAM (random access memory), que constitui quase que a totalidade da memória principal de um computador. O outro tipo, chamado de não volátil, não permite alterar ou apagar seu conteúdo, este tipo de memória conhecido como ROM (read-only memory), já vem pré-gravado do fabricante, geralmente com algum programa, e seu conteúdo é preservado mesmo quando a alimentação é desligada. Uma variação da ROM é a EPROM (erasable programmable ROM), onde podemos gravar e regravar a memória através exposição de luz ultravioleta por um dispositivo especial.

Atualmente, uma série de memórias com diferentes características, existe para diversas aplicações, como a EEPROM, EAROM, EAPROM, NOVRAM entre outras.

4.2 Hierarquia de memórias



Relação entre os diversos tipos de dispositivos de armazenamento.

4.2.1 Registradores

Os registradores são dispositivos de alta velocidade, localizados fisicamente na UCP para armazenamento temporário de dados. O número de registradores varia em função da arquitetura de cada processador. Alguns registradores são de uso específico e têm propósitos especiais, enquanto outros são ditos de uso geral.

Entre os registradores de uso específico, merecem destaque:

- contador de instruções (CI) ou program counter (PC) e o registrador responsável pelo armazenamento do endereço da próxima instrução que a UCP deverá executar. Toda vez que a UCP execute uma instrução, o PC é atualizado com um novo endereço;
- o apontador da pilha (AP) ou stack pointer (SP) e o registrador que contém o endereço de memória do topo da pilha, que é a estrutura de dados onde o sistema mantém informações sobre tarefas que estavam sendo processadas e tiveram que ser interrompidas por algum motivo;
- o registrador de estado, também chamado em alguns equipamentos de program status word (PSW), e o registrador responsável por armazenar informações sobre a execução do programa, como a ocorrência de carry e overflow. A cada instrução executada, o registrador de estado é alterado conforme o resultado gerado pela instrução.

O tamanho da capacidade de armazenamento dos registradores vai determinar o tamanho da palavra da arquitetura (exemplo 32 bits ou 64 bits).

4.2.2 Memória Cache

A memória cache é uma memória volátil de alta velocidade. O tempo de acesso a um dado nela contido é muito menor que se o mesmo estivesse na memória principal.

Toda vez que o processador fez referência a um dado armazenado na memória principal, ele "olha" antes na memória cache. Se o processador encontrar o dado na cache, não há necessidade do acesso a memória principal; do contrário, o acesso é obrigatório. Neste último caso, o processador, a partir do dado referenciado, transfere um bloco de dados para a cache. O tempo de transferência entre as memórias é pequeno se comparado com o aumento do desempenho obtido com a utilização desse tipo de memória.

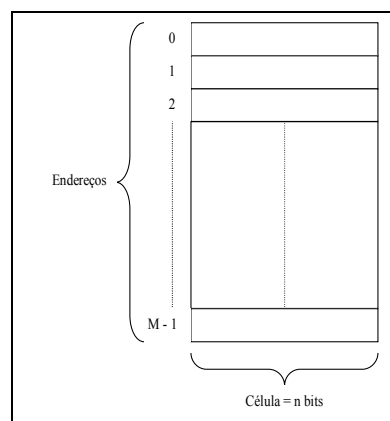
Apesar de ser uma memória de acesso rápido seu uso é limitado em função do alto custo.

Atualmente a memória cache pode ser encontrada acoplada a periféricos, como por exemplo, placas de vídeo.

4.2.3 Memória Principal

A memória principal, também conhecida como memória primária ou real, é a parte do computador onde são armazenadas instruções e dados. Ela é composta por unidades de acesso chamadas de células sendo cada célula composta por um determinado número de bits (binary digit). O bit é a unidade básica de memória podendo assumir o valor 0 ou 1. Atualmente a grande maioria dos computadores utiliza o byte (8 bits) como tamanho de célula, porém encontramos computadores de gerações passadas com células de 16, 32 e até mesmo 60 bits.

Podemos concluir, então, que a memória, formada por um conjunto de células, onde cada célula possui um determinado número de bits.



Memória principal

O acesso ao conteúdo de uma célula é realizado através da especificação de um número chamado endereço. O endereço é uma referência única que podemos fazer a uma célula de memória. Quando um programa deseja ler ou escrever um dado em uma célula deve primeiro especificar qual o endereço de memória desejado para depois realizar a operação.

A especificação do endereço é realizada através de um registrador denominado registrador de endereço de memória (memory register address - MAR), através do conteúdo deste registrador a unidade de controle sabe qual a célula de memória que será acessada. Outro registrador usado em operações com a memória é o registrador de dados da memória (memory buffer register—MBR), este registrador é utilizado para guardar o conteúdo de uma ou mais células de memória após uma operação de leitura ou para guardar o dado que será transferido para a memória em uma operação de gravação.

Operação de Leitura	Operação de gravação
<ol style="list-style-type: none"> 1. A UCP armazena no MAR, o endereço da célula a ser lida. 2. A UCP gera um sinal de controle para a memória principal, indicando que uma operação de leitura deve ser realizada. 3. O conteúdo da(s) célula(s), identificada(s) pelo endereço contido no MAR, e transferido para o MBR. 	<ol style="list-style-type: none"> 1. A UCP armazena no MAR, o endereço da célula que será gravada. 2. A UCP armazena no MBR, a informação que deverá ser gravada. 3. A UCP gera um sinal de controle para a memória principal, indicando que uma operação de gravação deve ser realizada. 4. A informação contida no MBR e transferida para a célula de memória endereçada pelo MAR.

Ciclo de leitura e gravação

4.2.4 Memória Secundária

A memória secundária é um meio permanente (não volátil) de armazenamento de programas e dados. Enquanto a memória principal precisa estar sempre energizada para manter suas informações, a memória secundária não precisa de alimentação.

O acesso a memória secundária é lento, se comparado com o acesso a memória cache ou à principal, porém seu custo é baixo e sua capacidade de armazenamento é bem superior a da memória principal. Enquanto a unidade de acesso a memória secundária é da ordem de milissegundos, o acesso a memória principal é de nanossegundos. Podemos citar, como exemplos de memórias secundárias, o disco magnético e os dispositivos de armazenamento.

4.3 Endereçamento Virtual

O endereço virtual é estruturado em duas partes, correspondentes à página virtual e ao byte dentro da página. O virtual page number (VPN) indica a página virtual em que o endereço está contido e o campo de deslocamento indica qual o byte especificado dentro da página.

O espaço de endereçamento virtual de cada processo é dividido em duas regiões denominadas região do processo e região do sistema. O bit mais significativo do endereço virtual indica se o endereço pertence à região do processo ou à do sistema.

A região do sistema é compartilhada por todos os processos. Neste caso, quando um processo faz referência a um endereço virtual dessa região, este tem o mesmo significado para todos os processos. Isto significa que quando dois processos fazem referência a um mesmo endereço virtual da região do sistema, eles estarão referenciando o mesmo código ou dado. Todo código do sistema operacional compartilhado entre processos está localizado nesta

região. Para que este compartilhamento seja realizado de forma organizada, o sistema implementa mecanismos de proteção para restringir o acesso às páginas.

Tanto a região do processo quanto a região do sistema são subdivididas em duas regiões, totalizando quatro regiões no espaço de endereçamento virtual. Cada uma dessas sub-regiões representa um tamanho de endereçamento. As duas sub-regiões do espaço do processo são denominadas, respectivamente, região do programa (P0) e região de controle (P1). Quando se executa um programa do usuário, o módulo executável é alocado em endereços virtuais da região P0, começando no endereço 0 e crescendo para os endereços maiores. Já na região P1, os endereços virtuais são alocados na seqüência dos maiores para os menores, em vista de tal região ser utilizada por estruturas de pilhas.

A região do sistema também é dividida em duas sub-regiões de tamanho determinado (S0 e S1). Os endereços virtuais da sub-região S0 contêm o código e estruturas do sistema operacional, como rotinas de interrupções, rotinas do sistema e tabelas de páginas. A sub-região S1 é reservada para uso futuro. Atualmente, a UCP não traduz um endereço virtual localizado nesta região. Caso algum processo faça referência a um endereço dentro dessa faixa, um erro de hardware ocorrerá.

4.4 Swapping

A paginação é um procedimento natural do mecanismo de memória virtual. Os processos estão permanentemente paginando e, para isso, o sistema operacional deve possuir sempre páginas disponíveis na memória principal para serem utilizadas. Em função disso, a lista de páginas livres deve sempre possuir uma quantidade mínima de páginas necessária para a utilização dos processos.

Em determinadas situações, a memória principal pode estar sendo bastante utilizada, levando a lista de páginas livres a valores baixos. Nesta situação específica, o sistema seleciona um ou mais processos e retira temporariamente o(s) working set(s) da memória principal, gravando-o(s) em disco (arquivo de swap). Com isso, páginas da memória principal são desocupadas, permitindo que o mecanismo da paginação continue seu funcionamento.

5. Gerência de dispositivos

Cada periférico do computador possui suas peculiaridades, assim, o procedimento de interação com uma placa de rede é completamente diferente da interação comum disco rígido SCSI. Todavia, existem muitos problemas e abordagens em comum para o acesso aos periféricos. Por exemplo, é possível criar uma abstração única para a maioria dos dispositivos de armazenamento como pen-drives, discos SCSI ou IDE, disquetes e etc, na forma de um vetor de blocos de dados. A função da gerência de dispositivos (também conhecida como gerência de entrada/saída) é implementar a interação com cada dispositivo por meio de drivers e criar modelos abstratos que permitam agrupar vários dispositivos distintos sob a mesma interface de acesso.

5.1 Dispositivos de Entrada e Saída

Os dispositivos de entrada e saída (E/S) são utilizados para permitir a comunicação entre o computador e o mundo externo. Através desses dispositivos a UCP e a memória principal podem se comunicar tanto com usuários quanto com memórias secundárias, a fim de realizar qualquer tipo de processamento.

Os dispositivos de E/S podem ser divididos em duas categorias: os que são utilizados como memória secundária e os que servem para a interface homem-máquina.

Os dispositivos utilizados como memória secundária como discos se caracterizam por armazenar grande volume de informações, seu custo é relativamente baixo e seu tempo de acesso é maior que o acesso a memória principal.

Alguns dispositivos servem para a comunicação homem-máquina, como teclados, monitores de vídeo, impressoras, entre outros. Com o avanço no desenvolvimento de aplicações de uso cada vez mais geral procura-se aumentar a facilidade de comunicação entre o usuário e o computador. A implementação de interfaces mais amigáveis permite cada vez mais que pessoas sem conhecimento específico sobre informática possam utilizar o computador. Scanner, caneta ótica, mouse, dispositivos sensíveis a voz humana e ao calor do corpo humano são alguns exemplos desses tipos de dispositivos.

5.2 Barramento

A UCP, a memória principal e os dispositivos de E/S são interligados através de linhas de comunicação denominadas barramentos, barras ou vias. Um barramento (bus) é um conjunto de fios paralelos (linhas de transmissão) onde trafegam informações, como dados, endereços ou sinais de controle. Ele pode ser classificado como unidirecional (transmissão em um só sentido) ou bidirecional (transmissão em ambos os sentidos).

Na ligação entre UCP e memória principal podemos observar que três barramentos são necessários para que a comunicação seja realizada. O barramento de dados transmite informações entre a memória principal e a UCP. O barramento de endereços é utilizado pela UCP para especificar o endereço da célula de memória que será acessada. Finalmente o barramento de controle é por onde a UCP envia os pulsos de controle relativos as operações de leitura e gravação.

6. Gerenciamento do processador

A gerência do processador implementada define a política de divisão do tempo do processador entre os processos dos usuários e do sistema operacional. O escalonamento de processos é realizado por uma rotina de interrupção do sistema denominada scheduler.

A política de escalonamento é implementada através de prioridades associadas aos processos, denominadas prioridades base, esta prioridade é estabelecida no momento da criação do processo.

6.1 Escalonamento de Tempo Compartilhado

Um processo em estado corrente (CUR), trabalhando na faixa de tempo compartilhado, somente deixa a UCP caso ocorra uma destas situações:

- término de execução da imagem;
- processo de maior prioridade entra em estado de COM (preempção por prioridade);
- solicitação de um evento ou recurso do sistema;
- término da fatia de tempo (prioridade por tempo).

Para o escalonamento de tempo compartilhado além da prioridade base definida na criação do processo, existe uma outra, chamada dinâmica, que varia de acordo com as características de execução do processo. O escalonamento de tempo compartilhado é realizado com base na prioridade dinâmica dos processos.

A prioridade dinâmica é alterada quando um processo sai do estado de espera para o estado de pronto. O sistema incrementa um valor à prioridade base em função do tipo de espera a que o processo estava submetido. Eventos que exigem longo tempo de espera

incidem em um incremento maior, com isso, um processo CPU-bound tende a ter uma prioridade dinâmica menor que a de um processo I/O-bound. Este esquema permite balancear o uso do processador entre todos os tipos de processos.

A prioridade dinâmica é calculada pela soma da prioridade base com o incremento recebido. Seu valor é decrementado ao longo do tempo, porém nunca poderá cair abaixo da prioridade base estabelecida.

6.2 Escalonamento de Tempo Real

Um processo em estado corrente (CUR), trabalhando na faixa de tempo real, somente deixa a UCP caso ocorra uma das seguintes situações:

- término de execução da imagem;
- processo de maior prioridade entra em estado de COM (preempção por prioridade);
- solicitação de um evento ou recurso do sistema.

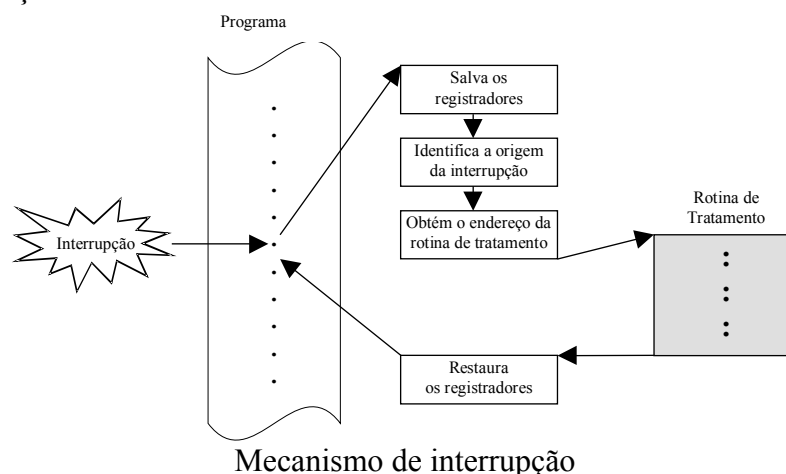
Existem duas diferenças na política de escalonamento para esses tipos de processos: a não-existência do conceito de fatia de tempo e de prioridade dinâmica.

6.3 Interrupções e Exceção

Quando um controlador de periférico tem uma informação importante a fornecer ao processador ele tem duas alternativas de comunicação:

- Aguardar até que o processador o consulte, o que poderá ser demorado caso o processador esteja ocupado com outras tarefas (o que geralmente ocorre);
- Notificar ao processador através do barramento de controle, enviando a ele uma requisição de interrupção (IRQ – Interrupt ReQuest).

Ao receber a requisição de interrupção o processador suspende seu fluxo de execução corrente e desvia para um endereço pré-definido onde se encontra uma rotina de tratamento de interrupção (interrupt handler). Essa rotina é responsável por tratar a interrupção, ou seja, executar as ações necessárias para atender o dispositivo que a gerou. Ao final da rotina de tratamento da interrupção o processador retoma o código que estava executando quando recebeu a requisição.



7. Gerenciamento de Proteção

Com computadores conectados em rede e compartilhados por vários usuários é importante definir claramente os recursos que cada usuário pode acessar, as formas de acesso permitidas (leitura, escrita, etc) e garantir que essas definições sejam cumpridas. Para proteger os recursos do sistema contra acessos indevidos é necessário:

- a) definir usuários e grupos de usuários;

b) identificar os usuários que se conectam ao sistema, através de procedimentos de autenticação;

c) definir e aplicar regras de controle de acesso aos recursos, relacionando todos os usuários, recursos e formas de acesso e aplicando essas regras através de procedimentos de autorização; e finalmente

d) registrar o uso dos recursos pelos usuários, para fins de auditoria e contabilização.

As funcionalidades do sistema operacional geralmente são inter-dependentes: por exemplo, a gerência do processador depende de aspectos da gerência de memória, assim como a gerência de memória depende da gerência de dispositivos e da gerência de proteção.

8. Gerenciamento de arquivos

Esta funcionalidade é construída sobre a gerência de dispositivos e visa criar arquivos e diretórios, definindo sua interface de acesso e as regras para seu uso. É importante observar que os conceitos abstratos de arquivo e diretório são tão importantes e difundidos que muitos sistemas operacionais os usam para permitir o acesso a recursos que nada tem a ver com armazenamento.

Exemplos disso são as conexões de rede (nos sistemas UNIX e Windows, cada socket TCP é visto como um descritor de arquivo no qual pode-se ler ou escrever dados) e as informações do núcleo do sistema (como o diretório /proc do UNIX).

8.1 Sistemas de arquivos

Um sistema de arquivos é um conjunto de estruturas lógicas e de rotinas, que permitem ao sistema operacional controlar o acesso ao disco rígido. Diferentes sistemas operacionais usam diferentes sistemas de arquivos. Conforme cresce a capacidade dos discos e aumenta o volume de arquivos e acessos esta tarefa torna-se mais e mais complicada, exigindo o uso de sistemas de arquivos cada vez mais complexos e robustos. Existem diversos sistemas de arquivos diferentes, que vão desde sistemas simples como o FAT16, que utilizamos em cartões de memória, até sistemas como o NTFS, EXT3 e ReiserFS, que incorporam recursos muito mais avançados.

Nos sistemas operacionais da Microsoft, temos apenas três sistemas de arquivos: FAT16, FAT32 e NTFS. O FAT16 é o mais antigo, usado desde os tempos do MS-DOS, enquanto o NTFS é o mais complexo e atual. Apesar disso, temos uma variedade muito grande de sistemas de arquivos diferentes no Linux (e outros sistemas Unix), que incluem o EXT2, o EXT3, o ReiserFS, o XFS, o JFS e muitos outros.

8.2 Partição e particionamento

A memória secundária possui muitos bytes, mas que são inúteis até que seja criada alguma tabela de alocação. É possível definir para uma unidade de memória secundária mais de uma tabela de alocação (do mesmo tipo ou de tipos distintos). Isso é feito dividindo-se a memória secundária em vários pedaços, chamados justamente de partições. Cada partição funcionará como um volume à parte. Partições diferentes permitem o uso em um mesmo computador de mais de um sistema operacional. As tabelas de alocação são guardadas em uma tabela de partições e definem o respectivo sistema de arquivos.

Particionar uma memória secundária significa dividi-la em várias partes, este é um procedimento necessário para que a memória secundária se torne funcional, sendo obrigatório a criação de no mínimo uma partição. Quando a memória secundária é particionada, automaticamente gera-se uma tabela de partições, onde fica gravado o endereço e a característica da partição gerada, as partições possuem características individuais para cada

tipo de sistema de arquivos. No sistema operacional Windows e MS-DOS são chamadas de FAT16 ou FAT32 o tipo de sistema de arquivo para a partição, em quanto que no Windows NT o sistema de arquivo para a partição pode ser do tipo NTFS e no Linux este é denominado EXT2, existem vários outros tipos sistema de arquivo para a partição usadas por outros sistemas operacionais. Depois de gerar a partição torna-se necessário formatá-la, este procedimento é feito através de um comando específico do sistema operacional que será utilizado no disco, no caso do MS-DOS usamos o comando "Format" para dar forma a partição, possibilitando a instalação do sistema operacional em questão.

8.2.1 Tabela de partições

No início da memória secundária existe uma área reservada destinada a tabela de particionamento, esta tabela informa ao sistema operacional de que forma o disco esta dividido, quais os tipos de sistema de arquivo das partições e também em que parte da memória secundária elas estão localizadas as partições assim como os seus tamanhos.

8.3 Volume

Conceito utilizado pelos sistemas operacionais da Microsoft que oferece uma visão abstrata de uma unidade de armazenamento, ou seja, área de armazenamento em uma memória secundária que é representada por uma letra (c:\; d:\ e etc) e geralmente representa uma partição, uma unidade de disco; um dispositivo de armazenamento; uma conexão de rede ou até mesmo um conjunto de unidades de armazenamento, neste caso é denominado volume distribuído.

8.4 formatação e formatos

A formatação física consiste na divisão dos discos magnéticos do HD (ou outro dispositivo de armazenamento) em trilhas, setores e cilindros e são gravadas as marcações servo, que permitem que a placa lógica posicione corretamente as cabeças de leitura. Nos HDs atuais, a formatação física é feita em fábrica, durante a fabricação dos discos. O processo envolve o uso de máquinas especiais e, apenas para garantir, restrições são adicionadas no firmware do drive, para que a placa lógica seja realmente impedida de fazer qualquer modificação nas áreas reservadas. Graças a isso, é impossível reformatar fisicamente um drive atual, independentemente do software usado.

A formatação lógica, que adiciona as estruturas utilizadas pelo sistema operacional. Ao contrário da formatação física, ela é feita via software e pode ser refeita quantas vezes você quiser. O único problema é que, ao reformatar o HD, você perde o acesso aos dados armazenados, embora ainda seja possível recuperá-los usando as ferramentas apropriadas.

8.4.1 FAT

File Allocation Table. Num HD armazena a lista dos endereços ocupados por cada arquivo guardado, permitindo localiza-los. A função da FAT é servir como um índice, armazenando informações sobre cada cluster do disco. Através da FAT, o sistema operacional sabe se uma determinada área do disco está ocupada ou livre, e pode localizar qualquer arquivo armazenado. Cada vez que um novo arquivo é gravado ou apagado, o sistema operacional altera a FAT, mantendo-a sempre atualizada.

A FAT é tão importante, que além da tabela principal, é armazenada também uma cópia de segurança, que é usada sempre que a tabela principal é danificada de alguma maneira. Uma curiosidade é que, quando formatamos um disco rígido usando o comando Format, nenhum dado é apagado, apenas a FAT principal é substituída por uma tabela em branco. Até que sejam reescritos porém, todos os dados continuam lá, podendo ser recuperados através de programas como o Lost & Found (Power Quest) ou o Easy Recovery (Ontrack).

8.4.2 NTFS

New Technology File System. O NTFS é um sistema de arquivos mais antigo do que muitos acreditam. Ele começou a ser desenvolvido no início da década de 1990, quando o projeto do Windows NT dava os seus primeiros passos.

Já que o grande problema do sistema FAT16 era o fato de serem usados apenas 16 bits para o endereçamento de cada cluster, permitindo apenas 65 mil clusters por partição, o NTFS incorporou desde o início a capacidade para endereçar os clusters usando endereços de 64 bits. A única limitação agora passa a ser o tamanho dos setores do HD. Como cada setor possui 512 bytes, o tamanho de cada cluster usando NTFS também poderá ser de 512 bytes, independentemente do tamanho da partição.

É sem dúvida um grande avanço sobre os clusters de 32 KB e as partições de até 2 GB da FAT 16. Mas, existe um pequeno problema em endereçar partições muito grandes usando clusters de 512 bytes: o desempenho. Com um número muito grande de clusters, o processamento necessário para encontrar os dados desejados passa a ser muito grande, diminuindo a performance.

Assim como na FAT 32, ficou estabelecido que o tamanho mínimo de clusters seria usado por default apenas em partições de um certo tamanho:

Tamanho da partição	Tamanho do cluster
até 512 MB	512 bytes
até 1 GB	1 KB
até 2 GB	2 KB
acima de 2 GB	4 KB

Apesar do default ser usar clusters de 4 KB em qualquer partição maior do que 2 GB, você pode criar partições com clusters do tamanho que desejar através do assistente para criação de partições do Windows 2000/XP, que pode ser encontrado em Painel de controle > Ferramentas Administrativas > Gerenciamento do computador > Armazenamento > Gerenciamento de disco. Do lado direito da tela será mostrado um mapa dos HDs instalados na máquina, basta clicar com o botão direito sobre uma área de espaço livre e em seguida em "criar partição":

Continuando, mais uma vantagem do sistema NTFS é que os nomes de arquivos e pastas utilizam caracteres em Unicode, em vez de ACSII. O ASCII é o sistema onde cada caracter ocupa 1 byte de dados, mas são permitidas apenas letras, números e alguns caracteres especiais. No Unicode, cada caracter ocupa dois bytes, o que permite 65 mil combinações, o suficiente para armazenar caracteres de diversos idiomas. Isso permite que usuários do Japão, China, Taiwan e outros países que não utilizam o alfabeto ocidental, possam criar arquivos usando caracteres do seu próprio idioma, sem a necessidade de instalar drivers e programas adicionais que adicionem o suporte.

Outro ponto importante onde o NTFS é superior ao sistema FAT é na tolerância a falhas. No sistema FAT, sempre que o sistema trava ou é desligado enquanto estão sendo atualizados arquivos os diretórios no HD, existe uma possibilidade muito grande do sistema tornar-se inconsistente, com arquivos interligados, agrupamentos perdidos e os outros problemas. Surge então a necessidade de rodar o scandisk depois de cada desligamento incorreto.

No NTFS, o sistema mantém um log de todas as operações realizadas. Com isto, mesmo que o micro seja desligado bem no meio da atualização de um arquivo, o sistema poderá, durante o próximo boot, examinar este log e descobrir exatamente em que ponto a atualização parou, tendo a chance de automaticamente corrigir o problema. Além de reduzir a perda de tempo, a possibilidade de perda de dados é muito menor.

Clusters contendo setores defeituosos também são marcados automaticamente, conforme são detectados, sem a necessidade de usar o scandisk ou qualquer outro utilitário.

Neste caso, a marcação é feita na tabela de endereçamento da partição, de forma que a lista de setores defeituosos é perdida ao reparticionar o HD. Antigamente, os HDs eram menos confiáveis e o aparecimento de setores defeituosos um fenômeno muito mais comum, de forma que a maioria dos programas de formatação realizavam um teste de superfície durante a formatação da partição (como no caso do format usado no Windows 95/98, onde formatar uma partição podia demorar mais de uma hora). Atualmente, a maioria dos programas realiza uma formatação rápida, presumindo que o HD não possua setores defeituosos.

Existiram diversas versões do NTFS, que acompanharam a evolução do Windows NT. A partir do Windows 2000, foi introduzido o NTFS 5, que trouxe diversos aperfeiçoamentos, incluindo o suporte ao Active Directory.

Outro recurso interessante é a possibilidade de encriptar os dados gravados, de forma a impedir que sejam acessados por pessoas não autorizadas mesmo caso o HD seja removido e instalado em outro micro. Este recurso de encriptação é interessante, por exemplo, para profissionais de campo, que levam dados secretos em seus laptops. É possível tanto criptografar o disco inteiro, quanto pastas ou arquivos individuais.

Também é possível compactar pastas e arquivos individuais, economizando espaço em disco. No Windows 95/98 era possível compactar partições usando o drvspace, mas só era possível compactar partições inteiras, o que normalmente acaba não sendo um bom negócio, pois diminuía bastante a velocidade do micro e aumentava a possibilidade de perda de dados. Naturalmente, a compactação também é diferente da feita por programas como o Winzip, já que os arquivos e pastas continuam acessíveis exatamente da mesma forma, com o sistema fazendo a compactação e descompactação do arquivo de forma transparente.

Com a possibilidade de compactar pastas individuais, você pode comprimir apenas as pastas contendo um grande volume de arquivos que suportam um bom nível de compressão, deixando de lado pastas com fotos, músicas e arquivos de vídeo, arquivos que já estão comprimidos. Para compactar uma pasta, acesse o menu de propriedades. Na seção "avançadas", marque a opção de compactar arquivos para economizar espaço.

A compactação de arquivos exige uma carga adicional de processamento, já que o sistema tem o trabalho de descompactar os arquivos antes de acessá-los. Antigamente, usar compactação reduzia muito o desempenho do sistema, já que os processadores eram mais lentos. Num micro atual, a redução é muito menos significativa e, em muitos casos, o uso da compactação pode até mesmo melhorar o desempenho, já que arquivos compactados ocupam menos espaço e, conseqüentemente, são lidos mais rapidamente pela cabeça de leitura.