

VI - Introdução aos Sistemas Operacionais

Consiste de um ou um conjunto de programas que compõem o software básico do computador e cuja finalidade é a de executar os programas aplicativos e de servir de interface entre o computador e seus usuários. Um sistema operacional deve atender a três objetivos principais:

- a) Conveniência - tornar o uso do computador mais conveniente (fácil).
- b) Eficiência - tornar eficiente (seguro e justo) o uso e o compartilhamento dos recursos existentes
- c) Evolução - possibilitar o constante *debug* e o desenvolvimento de novas funcionalidades

Numa abordagem macro, o sistema operacional pode ser visto como a primeira camada de software acima do hardware do computador que, conforme mostra a figura VI.1 abaixo, se encarrega de suportar e servir de interface entre este e os demais programas aplicativos e utilitários.

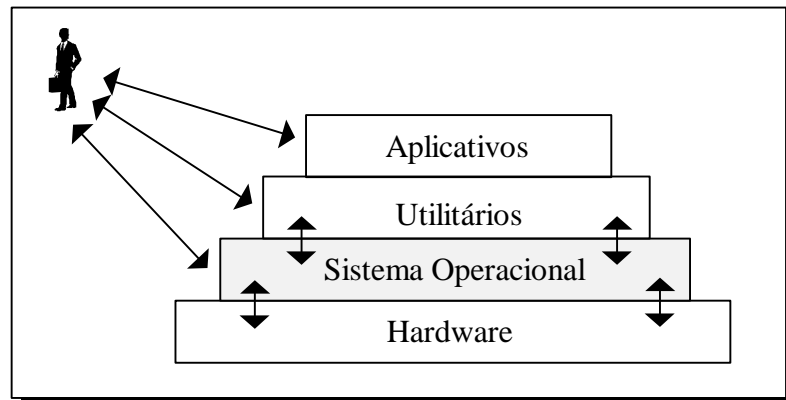


Figura VI.1 - Visão do Sistema por Camadas

Um sistema operacional hoje se constitui de diversos módulos que se encarregam da comunicação, alocação e gerenciamento de recursos específicos tais como:

- Processo
- Memória
- Arquivos
- Entrada e Saída
- Interconexão
- Alocação da UCP ("Scheduling")
- Segurança e
- Interface com o Usuário

A interface entre os programas aplicativos (processos) e o sistema operacional é realizada através de comandos (instruções) de chamada do sistema (as *system calls*). Estas chamadas estão geralmente disponíveis ao nível das linguagens assembly e, em geral, são introduzidas nos programas escritos em linguagens de mais alto nível, pelos respectivos compiladores.

Grande parte dos programadores não tomam conhecimento das *system calls*, embora seus programas façam uso intenso delas. O fato é que muitos dos detalhes da interface com os

sistemas operacionais são escondidos do programador pelos compiladores e pelo sistema de suporte a execução dos processos.

As *system calls* podem ser genericamente grupadas em 5 categorias:

- a) controle de processo
 - finalizar / abortar
 - carregar / executar
 - criar / terminar
 - pegar / setar atributos
 - esperar determinado tempo
 - esperar por evento / sinalizar evento
 - alocar / liberar memória
- b) manipulação de arquivos
 - criar / eliminar arquivos
 - abrir / fechar
 - ler / escrever
 - pegar / setar atributos
- c) manipulação de dispositivos
 - montar / liberar dispositivo
 - ler / escrever
 - pegar / setar atributos
 - anexar logicamente (attach) / liberar dispositivos
- d) informações de manutenção
 - pegar / setar hora e data
 - pegar / setar atributos de processos, arquivos e dispositivos
- e) comunicações
 - estabelecer / finalizar conexão
 - enviar / receber mensagens
 - transferir informações de estado
 - anexar logicamente / liberar dispositivos remotos

A história mostra que os sistemas operacionais vêm sofrendo constantes modificações com o passar do tempo. No início eram relativamente simples, do tipo monousuário e praticamente não ofereciam qualquer mecanismo de proteção aos usuários. Com a evolução do hardware e o surgimento de equipamentos cada vez mais velozes, os sistemas operacionais foram se tornando mais complexos, mais seguros, mais eficientes e mais abrangentes.

Diversos tipos de sistemas operacionais podem ser identificados: monoprogramáveis, multiprogramáveis, multiprocessáveis, sistemas em rede, sistemas distribuídos, sistemas em lote ("batch"), sistemas de tempo compartilhado ("time sharing") e de tempo real.

VI.1 - Sistemas Monoprogramáveis (ou monotarefa)

Se caracterizam pela execução de uma única tarefa (processo) por vez, sendo que todos os recursos (processador, memória e periféricos) ficam exclusivamente a ela dedicados. Nesses sistemas, enquanto o programa aguarda a ocorrência de um evento qualquer, o processador ficará ocioso ("idle"); a memória ficará subutilizada, caso o programa não a ocupe totalmente e os periféricos também ficarão ociosos se não utilizados.

Os processos, como mostrado na figura VI.2 abaixo, são executados em seqüência e um só inicia após o término do anterior. Os espaços em branco representam os períodos de ociosidade da UCP, enquanto aguarda a realização ou execução de um evento externo qualquer solicitado pelo programa.

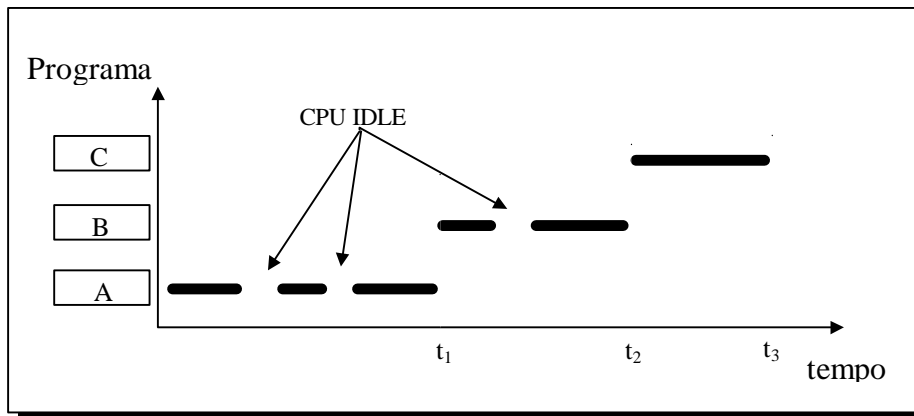


Figura VI.2 - Execução de 3 tarefas em um sistema monoprogramável

VI.2 - Sistemas Multiprogramáveis (multitarefa)

Se caracterizam por permitir que vários programas (tarefas) *residam simultaneamente na memória e concorram* pelo uso dos recursos disponíveis (apenas um programa detém, num determinado instante, o controle da UCP). São muito mais complexos e eficientes que os sistemas monoprogramáveis.

Nestes Sistemas, enquanto uma tarefa aguarda a ocorrência de um evento externo a UCP, esta pode atender outra tarefa qualquer, que esteja em condições de ser executada. O sistema operacional se encarrega de gerenciar o acesso concorrente das diversas tarefas aos diversos recursos, de forma ordenada e protegida. Como pode ser visto na figura VI.3, o *throughput* do sistema melhora, isto é, o número de processos concluídos por unidade de tempo aumenta, embora o tempo de execução de cada processo possa sofrer uma piora. Comparando com a execução monoprogramável nota-se que: $t_a > t_1$, $t_b - t_x > t_2 - t_1$, $t_c - t_y > t_3 - t_2$, porém $t_c < t_3$.

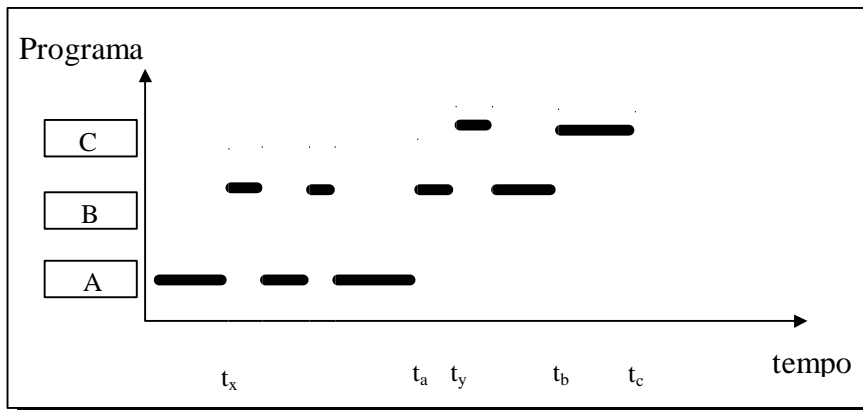


Figura VI.3 - Execução de 3 tarefas em um sistema multiprogramável

É imediato concluir que um sistema multiprogramável oferece condições de maior eficiência computacional que um sistema monoprogramável. Aproveitando os diagramas de tempo das figuras VI.2 e 3, podemos resumir o ganho de eficiência através do seguinte exercício:

Supondo que os processos A, B e C sejam dos tipos abaixo,

	A	B	C
Tipo do processo	I/O bounded	I/O bounded	UCP bounded
Duração	20 min	15 min	15 min
Memória requerida	40%	20%	30%
Forma de Processamento	batch	time-sharing	batch
Acesso a Disco	sim	não	não
Acesso a Impressora	não	sim	não

e considerando $t_x = 10$ min, $t_a = 25$ min, $t_y = 30$ min, $t_b = 35$ min, $t_c = 40$ min, I/O bounded = 20%, UCP bounded = 80% e os intervalos de UCP idle iguais a 2 minutos cada, teremos como resultado o seguinte desempenho do sistema na execução dos 3 processos:

	Monoprogramável	Multiprogramável
Uso do Processador (UCP)	35,6%	44,5%
Uso da Memória	31%	45%
Uso do Disco	40%	50%
Uso da Impressora	30%	37,5%
Taxa de Throughput	3,6 jobs/hr	4,5 jobs/hr
Elapsed Time	50 min	40 min
Tempo Médio de Resposta	35 min	33 min

onde:

- a) uso do processador é dado por -
 - $((20 - 4) \cdot 20 + (15 - 2) \cdot 20 + 15 \cdot 80) / 50 = 35,6\%$
 - $((20 - 4) \cdot 20 + (15 - 2) \cdot 20 + 15 \cdot 80) / 40 = 44,5\%$
- b) uso da memória -
 - $(20 \cdot 40 + 15 \cdot 20 + 15 \cdot 30) / 50 = 31\%$
 - $(10 \cdot 40 + (25 - 10) \cdot 60 + (30 - 25) \cdot 20 + (35 - 30) \cdot 50 + (40 - 35) \cdot 30) / 40 = 45\%$
- c) uso do disco -
 - $(20 \cdot 100) / 50 = 40\%$
 - $(20 \cdot 100) / 40 = 50\%$
- d) uso da impressora -
 - $(15 \cdot 100) / 50 = 30\%$
 - $(15 \cdot 100) / 40 = 37,5\%$
- e) throughput -
 - $3 \cdot 60 / 50 = 3,6$ jobs / hr
 - $3 \cdot 60 / 40 = 4,5$ jobs / hr
- f) elapsed time (tempo total de execução dos 3 processos)
- g) tempo médio de resposta -
 - $(20 + 35 + 50) / 3 = 35$ minutos
 - $(25 + 35 + 40) / 3 = 33,3$ minutos

Um sistema multiprogramável pode ser dos tipos: lote ("batch"), tempo compartilhado ("time sharing") ou tempo real ("real time"), sendo que um único sistema pode suportar um ou mais destes tipos de processamento.

- **Sistemas "Batch"**

Se caracterizam pela execução de programas previamente introduzidos e armazenados no computador. Não há interação com o usuário e os programas armazenados vão sendo executados na medida que haja disponibilidade de recursos.

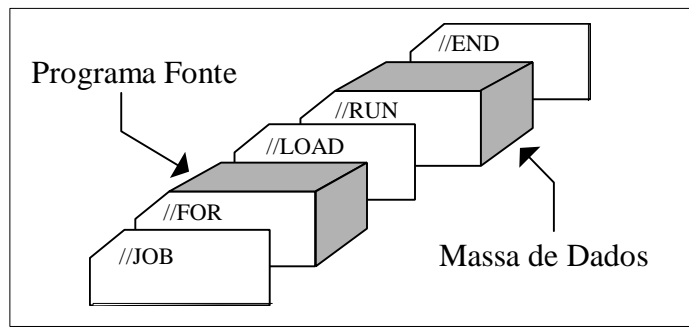


Figura VI.4 - Sistema "Batch" típico em cartões

- **Sistemas "Time Sharing"**

Surgiram com o aparecimento dos terminais de vídeo (terminais "burros") que permitiram ao usuário compartilhar à distância o uso dos recursos do computador. Dessa forma o usuário acessa e interage com o computador tanto na fase de desenvolvimento de suas aplicações como nas fases de execução e análise dos resultados.

Os terminais eram chamados "burros" (padrão TTY) por serem desprovidos de qualquer capacidade de processamento local, que era todo executado pela UCP do sistema central. Para que cada usuário tenha chances equilibradas de utilizar os recursos, o sistema aloca uma fatia de tempo ("time slice") do processador para cada terminal ativo e os atende num esquema de fila circular. Esgotado o "time slice" de um usuário, ele perde automaticamente o controle da UCP, que passa para o usuário seguinte da fila. Alguns sistemas permite a adoção de níveis de prioridade que altera a sequência de atendimento da fila ou o tamanho do "time slice" adotado para um usuário.

Não só o processador é compartilhado neste sistema, mas também a memória e os periféricos, como discos e impressoras. O sistema cria para o usuário um ambiente de trabalho próprio, dando a impressão de que todo o sistema está dedicado, exclusivamente a ele.

- **Sistemas de Tempo Real**

São estrutural e conceitualmente semelhantes aos sistemas multiprogramados, a diferença fundamental está no tempo de resposta exigido na execução (atendimento) das tarefas. Enquanto nos primeiros o tempo de resposta pode variar sem comprometer as aplicações em execução, nos sistemas de tempo real os tempos de resposta geralmente estão dentro de limites rígidos que, se não observados, podem inviabilizar a aplicação ou trazer problemas críticos de funcionamento.

Nos sistemas de tempo real não existe a idéia de fatia de tempo ("time slice"). Neles um programa mantém o controle da UCP pelo tempo que for necessário ou até que outro processo de maior prioridade apareça.

Sistemas de tempo real são comuns em aplicações de controle de processos tais como o monitoramento de refinarias de petróleo, de usinas termoelétricas e nucleares, controle de tráfego aéreo ou qualquer aplicação onde a fator tempo de resposta é crítico e fundamental.

- **Sistemas Mono e Multitarefa**

Uma outra terminologia mais recentemente introduzida no jargão da informática é a de sistemas monotarefa e multitarefa. O conceito por trás desta terminologia é subjetivo onde monotarefa é entendido como um sistema monoprogramável / monousuário, como nos sistemas PC com DOS, e multitarefa como um sistema multiprogramável / monousuário, como no sistema Linux, onde apenas um usuário faz uso do sistema porém podendo disparar diversas tarefas concorrentes.

VI.3 - Sistemas Multiprocessados

Caracterizam-se por permitir a execução *simultânea* de duas ou mais instruções, o que requer a existência de mais de um processador. O multiprocessamento mantém todos os conceitos da multiprogramação agora aplicados a vários processadores ao mesmo tempo.

O multiprocessamento pode ser obtido pela configuração de múltiplos processadores que compartilham de uma mesma memória primária (fortemente acoplados) ou de múltiplos computadores independentes do tipo sistemas em rede e sistemas distribuídos (fracamente acoplados), onde cada um tem seus próprios recursos.

Os sistemas multiprocessados permitem que vários programas sejam executados em paralelo (granularidade grossa), ou que um programa tenha duas ou mais de suas instruções executadas em paralelo (granularidade fina).

VI.4 - Sistemas em Rede

Se caracterizam pela existência de vários computadores independentes interligados em rede e compartilhando alguns recursos tais como disco, impressora, scanner e outros. Um sistema operacional de rede se encarrega de propiciar o protocolo para comunicação e transferência de dados entre os usuários e servidores da rede. Cada nó da rede é independente e capaz de executar sua própria aplicação.

VI.5 - Sistema Distribuído

É conceitualmente um sistema em rede que possibilita uma integração e uma cooperação transparente dos diversos nós que compõem a rede. Desta forma, sob o enfoque dos usuários e das tarefas, o sistema é uno e se comporta como uma arquitetura multiprocessada possibilitando tanto paralelismo de granularidade grossa como fina.

Sistemas 100% distribuídos ainda não estão comercialmente disponíveis, mas representam uma tendência natural e desejada para os atuais sistemas em rede.

VI.6 - Estrutura do Sistema Operacional

Podem ser dos tipos monolítico, em camadas (modular) ou cliente-servidor. (fig. VI.5)

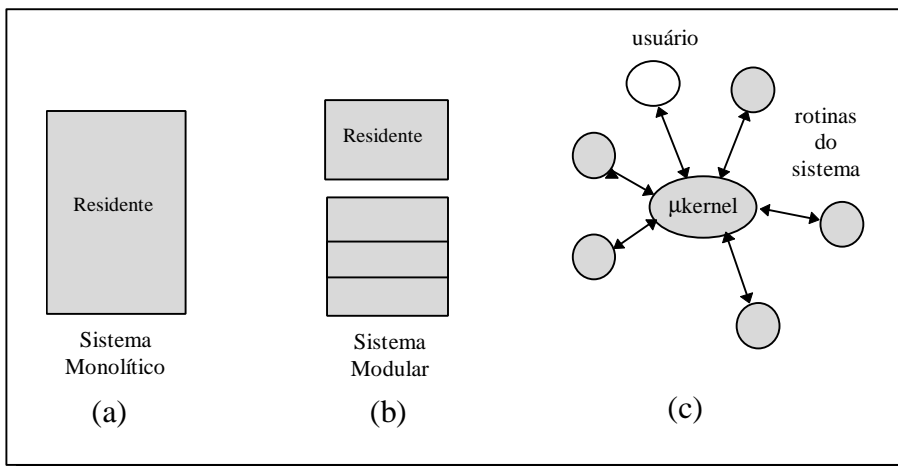


Figura VI.5 - Modelos de Configuração de um Sistema Operacional

O modelo monolítico (fig. VI.5a) imperou durante os primórdios da computação, nele o sistema operacional é escrito como um programa único composto por uma coleção de sub-rotinas que chamam umas as outras sempre que necessário.

Para efeito de segurança, mesmo os modelos monolíticos, adotam modos distintos de operação, no mínimo dois - modo usuário e modo supervisor (kernel), que operam com privilégios e prioridades distintas de execução. Aos programas aplicativos é reservado o modo usuário, com menos privilégio e menor prioridade, e às rotinas do sistema operacional é reservado o modo supervisor. Na figura VI.6 abaixo é mostrado como um programa no modo usuário acessa recursos do sistema através da chamada a rotinas do sistema (system calls).

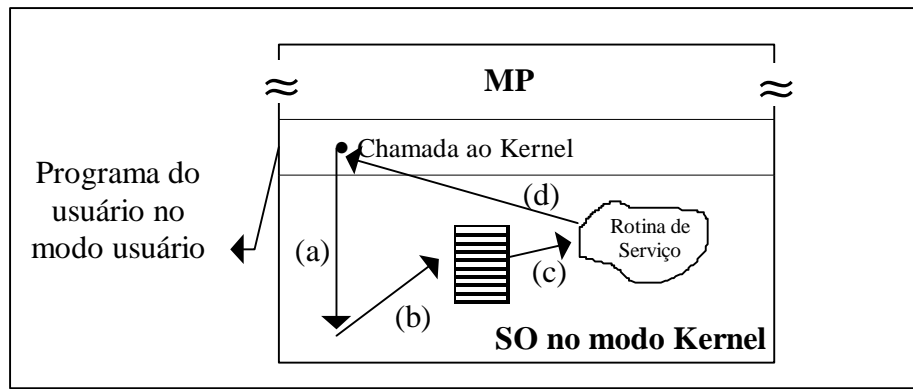


Figura VI.6 - Chamada de uma Rotina do Sistema (System Call): (a) o programa aplicativo realiza um trap para o Kernel; (b) o SO, através de uma tabela determina o endereço da rotina de serviço; (c) a rotina de serviço é acionada; (d) o serviço solicitado é executado e o controle retorna ao programa aplicativo

Na filosofia em camadas a comunicação ocorre apenas entre camadas adjacentes e, assim, fica estabelecida a segurança e o funcionamento do sistema. A camada mais inferior é a que tem acesso aos dispositivos de hardware e a camada mais externa é a que realiza a interface com o aplicativo do usuário.

Uma tendência nos sistemas operacionais modernos é a estruturação segundo a filosofia cliente-servidor. Nesta estratégia o *Kernel* tende a ser reduzido ao máximo, tornando-se as vezes um *micro-kernel*, e as tarefas passam a ser executadas por programas de sistema, chamados de *servidores*, que são executados no modo usuário. O programa aplicativo, agora chamado de *cliente*, para requisitar a execução de um serviço, por exemplo a leitura de um bloco de disco, envia uma mensagem ao processo servidor, que realiza a tarefa e envia de volta a resposta ao cliente.

Nesta estrutura o kernel se encarrega apenas de algumas funções consideradas básicas como, por exemplo: o mecanismo de intercomunicação entre processos, o gerenciamento de memória, o escalonamento de processos e o controle e gerenciamento de interrupção. Esta parte central do sistema operacional fica bastante reduzida em termos de tamanho e passa a ser denominada micro-kernel. Uma das principais vantagens desta estratégia é a sua fácil adaptabilidade para uso em sistemas distribuídos (fig. VI.7).

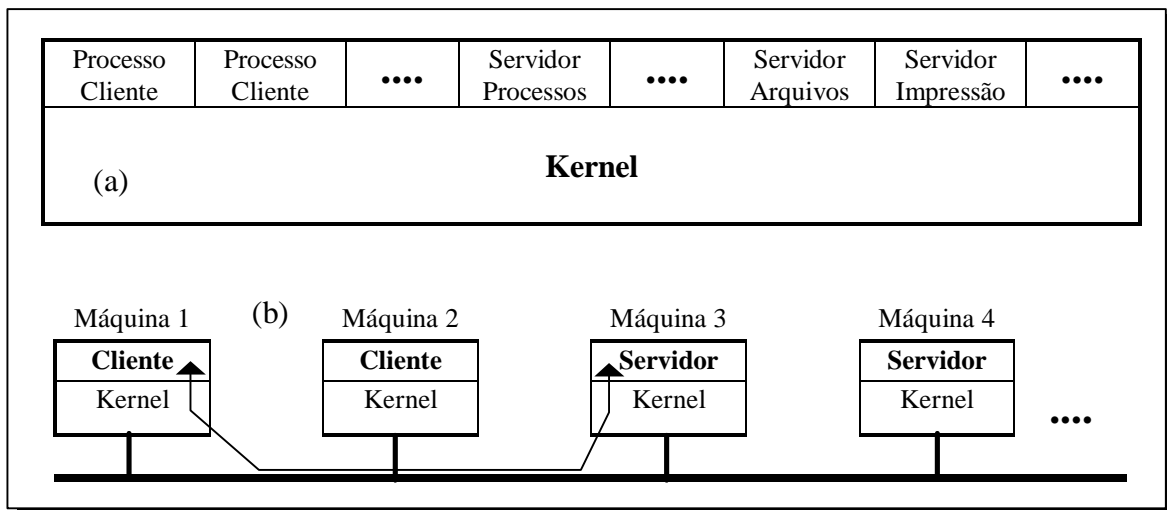


Figura VI.7 - Modelo Cliente-Servidor: (a) centralizado e (b) distribuído

VI.7 - Máquinas Virtuais

Nesta técnica uma máquina real pode abrigar internamente diferentes ambientes virtuais, cada um simulando uma máquina distinta, com memória, sistema operacional, recursos e processos próprios (fig. VI.8). Desta forma cada usuário ou aplicativo parece possuir sua própria máquina.

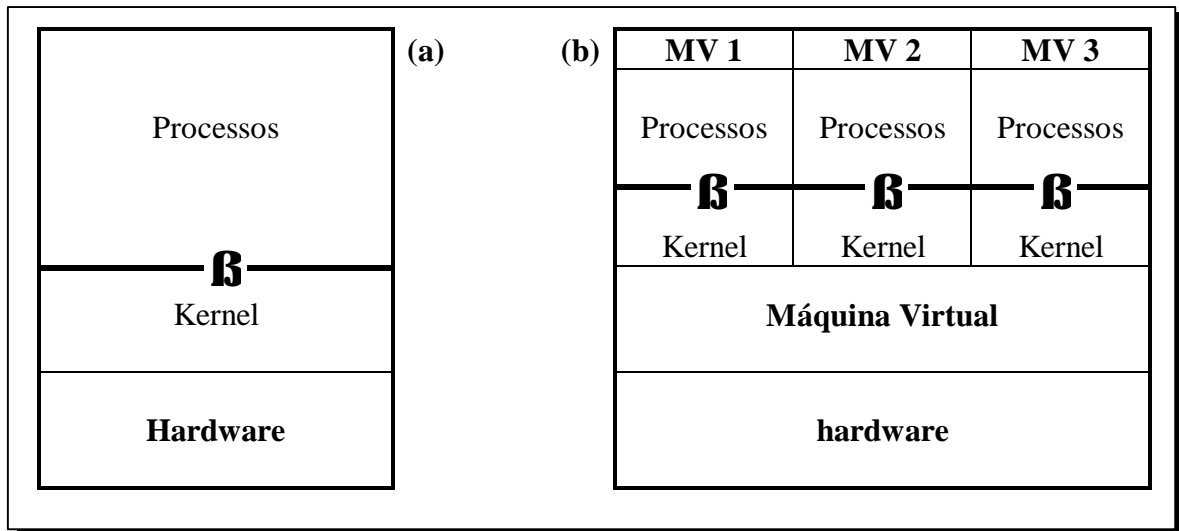


Figura VI.8 - Modelos de Sistemas: (a) Máquina Não-Virtual; (b) Virtual

VI.8 - Referências Bibliográficas

- Davis, William S., Sistemas Operacionais - Uma Visão Sistemática, Ed. Campus, 1990.
- Machado, Francis B. e Maia, Luiz P., *Introdução à Arquitetura de Sistemas Operacionais*, Ed. LTC, 1994.
- Stallings, William, Operating Systems, Ed. Acmillan Publishing Company, 1992.
- Silberschatz, Abraham e Galvin, Peter B., Operating Systems Concepts, Ed. Addison-Wesley Publishing Company, 1994.
- Tanenbaum, Andrew S., *Sistemas Operacionais Modernos*, Ed. Campus, 1995.