SO – Anotações

# Sistema Operacional:

É um programa ou um conjunto de programas interrelacionados cuja finalidade é atuar como: Intermediário (interface) entre o usuário e hardware, e gerenciador de recursos. 2 funções: fornece abstrações para os programas de usuários e gerencia os recursos do computador.

# Visões

Uma visão mais moderna trata SO como uma máquina virtual, além de ter 3 tipos:

Resource Manager – gerencia e aloca recursos;

Control program – controla a execução de programas usuários e operações de E/S;

Command Executer – fornece um ambiente para execução de comandos.

# Kernel

O sistema operacional é o único programa em execução o tempo todo no computador.

Programa de sistemas: geralmente associados ao sistema operacional, mas não necessariamente parte do kernel (núcleo);

Programa de aplicações: que inclui todos os programas não associado à operação do sistema.

* É possível interagir com o SO de diversas maneiras:
  + Usuário: linguagem de comando, interface gráfica;
  + Programa de usuário: Invoca o SO por meio de syscalls.
* Compartilhamento de HW:
  + Monoprogramado: monotarefa, somente um programa ativo por vez em dado período.
  + Multiprogramados: multitarefa, mais de um programa ao mesmo tempo por meio de compartilhamento de recurso (CPU, RAM etc.).

# SO Monotarefa

“Monoprogramado”. Permite que a CPU, memória e periféricos permaneçam exclusivamente dedicados a um único programa. Alocação de memória para apenas um processo, SO fica em uma camada protegida da memória.

Desvantagem: ociosidade do processador e subutilização da memória (processador fica ocioso quando o programa espera uma E/S).

Vantagem: simples implementação.

Exemplo: MS-DOS.

# SO Multitarefa

“Multiprogramado”. Vários programas em memória ao mesmo tempo e várias tarefas simultâneas. Logo, quando uma roda, a outra espera, e requer mecanismo de troca rápidas. Podem ser classificados de acordo com a forma com que suas aplicações são gerenciadas, podendo ser divididos em sistemas batch, de tempo compartilhado e de tempo real.

Desvantagem: implementação muito mais complexa.

Vantagem: redução do tempo de resposta das aplicações, além dos custos reduzidos devido ao compartilhamento dos recursos do sistema entre as diferentes aplicações.

Exemplos: Windows, Linux, Android.

# Multitarefas em Batch

Sistema em batch (lote), consiste em submeter ao computador um lote de programas de um só vez. Os jobs são submetidos em ordem sequencial para a execução e não existe interação entre o job e o usuário durante a execução.

# Multitarefas de tempo de compartilhamento

SO’s interativos. Onde o sistema permite que os usuários interajam com suas computações na forma de diálogo, e podem ser projetados como sistemas monousuários ou multiusuários (usando conceitos de multiprogramação e timesharing).

Timesharing: variante da multiprogramação, cada usuário tem um terminal online, onde o SO vê quando a CPU pode ser alocada para as tarefas que demandam serviços. Tem um intervalo de tempo fixo para os diferentes processos. Seu objetivo principal é o tempo de resposta interativo.

* + Vantagem: resposta rápida, reduz tempo ocioso da CPU, todas as tarefas recebem um tempo específico e menor probabilidade de duplicação de software.
  + Desvantagem: consumo de muitos recursos, requer alta especificação de hardware, confiabilidade, segurança, integridade e comunicação de dados.

Multiprogramação: divisão da memória em várias partes, com uma tarefa diferente em cada partição. Permite executar vários processos monitorando seus estados de processo e alternando entre os processos. Não tem intervalo de tempo fixo para processos. Seu objetivo principal é a utilização de recursos.

* + Vantagem: sem tempo ocioso da CPU, tarefas são executadas em paralelo, tempo de resposta mais curto, maximiza o rendimento total do computador, aumenta a utilização de recursos.
  + Desvantagem: trabalhos de longa duração têm que esperar muito tempo, rastreamento de todos os processos é difícil, requer agendamento de CPU, requer gerenciamento de memória eficiente, nenhuma interação do usuário com qualquer programa durante a execução.

# Multitarefas de tempo real

Usados para servir aplicações que atendem processos externos, e que possuem tempos de resposta limitados. Geralmente sinais de interrupções comandam a atenção do Sistema e geralmente são projetados para uma aplicação específica.

# Estruturas de SO’s

São 6: sistemas monolíticos, micronúcleos, sistemas de camadas, sistemas cliente-servidor, máquinas virtuais e exonúcleos.

# Estrutura monolítica

É a forma mais primitiva. Consiste em um conjunto de programas que executam sobre o hardware, como se fosse um único programa. Os programas de usuário podem ser vistos como subrotinas, invocadas pelo SO, quando este não está executando nenhuma das funções do sistema.

# Estrutura micronúcleo

“Microkernel”. Incorpora somente as funções de baixo nível mais vitais. Fornece uma base sobre a qual é construído o resto do SO. A maioria destes sistemas são construídos como coleções de processos concorrentes e fornece serviços de alocação de CPU e de comunicação aos processos (IPC). O micronúcleo controla interrupções, processos, escalonamento, IPC.

# Estrutura em camadas

Apresenta modularização, que é a divisão de um programa complexo em módulos de menor complexidade. Os módulos interagem através de interfaces bem definidas. E, Informação Escondida, que é quando os detalhes das estruturas de dados e algoritmos são confinados em módulos. Externamente, um módulo é conhecido por executar uma função específica sobre objetos de determinado tipo.

# Estrutura máquina virtual

Apresenta a hierarquia de níveis de abstração, de modo que, a cada nível, os detalhes de operação dos níveis inferiores possam ser ignorados. Através disso, cada nível pode confiar nos objetos e operações fornecidas pelos níveis inferiores.

Exemplo Hierarquia:

Nível 1 – núcleo, responsável por alocação de CPU aos processos, HW: CPU.

Nível 2 – E/S básico, responsável por processos de E/S, HW: dispositivos de E/S.

Nível 3 – gerenciamento de memória, HW: memória principal.

Nível 4 – sistema de arquivos, HW: armazenamento secundário.

Nível 5 – interpretador de linguagem de comando, de mensagem, HW: controle do operador.

O Modelo de Máquina Virtual, cria um nível intermediário entre o hardware e o SO, denominado Gerência de Máquinas Virtuais. Cria diversas máquinas virtuais independentes, onde cada uma oferece uma cópia virtual do hardware, incluindo modos de acesso, interrupções, dispositivos de E/S etc. Como é independente, cada máquina pode ter seu próprio SO.

# Chamadas de sistema

Syscall. Porta de entrada para o modo kernel. Altera entre modo usuário e modo kernel. Feitas através de TRAPS (interrupções de sw), e após o término da chamada, a execução continua.

Normalmente, tem uma interface para esconder a complexidade das syscalls e é fornecida pelo SO. Geralmente escrita em linguagem de alto nível. As aplicações utilizam uma API, que é uma interface que encapsula o acesso direto às syscalls.

Motivações: Portabilidade, esconder complexidade e acréscimo de funcionalidades que otimizam o desempenho.

# Interrupções

Interrupções não são causadas por aplicações em execução, diferente de TRAPS. São modificações no fluxo de controle de um programa causadas por um evento externo ao processamento do programa, usualmente eventos relacionados a operações de E/S. As interrupções também param o processamento do programa atual e transferem o controle para uma rotina de tratamento de interrupção.

3 Tipos: Hardware, Software, Interna (exceção).

* + HW: Evento externo;
  + SW: Execução de uma instrução específica. Pode ser utilizada pelo programador para interrupção em algum ponto do programa.
  + Interna (Exceção): Uso inválido ou errado de uma instrução ou dado.

# Traps

É uma espécie de chamada a procedimento automática iniciada sempre que ocorrer alguma condição específica causada pela execução de um programa. Quando há ocorrência de uma trap, o fluxo de controle é alterado para um endereço fixo de memória. Nesse endereço fixo há uma instrução de desvio para um procedimento, conhecido como procedimento de tratamento de trap.

Exemplo: Overflow

# Interrupções vs Traps

Traps são síncronas e Interrupções assíncronas. Traps acontecem sincronamente porque resultam da execução do próprio programa, interrupções podem ser causadas por agentes externos.

# Processos

Instância de um programa em execução, único. É a forma pela qual o SO vê um programa e o executa. Cada processo tem seu espaço de endereçamento. Cada processo tem seu program counter, pilha de memória (contexto de hw). Cada processo tem seu id, escalonados pelo núcleo (contexto de sw).

* + 1º plano (foreground): Interação direta com o usuário.
  + 2º plano (background): Processos com funções específicas que independem do usuário.

Daemons: Processos que ficam em segundo plano para lidar com algumas atividades, como e-mail, páginas da web, notícias, impressão etc.

Processos são criados quando:

* Inicia o Sistema
* syscall para a criação de um processo, realizado por algum processo em execução
* há requisição do usuário para a criação de um novo processo
* há inicialização de um processo em batch

Processos são finalizados quando:

* Normal - A tarefa é finalizada
* Erro - O processo sendo executado não pode ser finalizado
* Erro fatal – não encontrou o arquivo
* Outro processo.

Mudança de contexto: salvar o conteúdo dos registradores do processo que está deixando a CPU e carregar os reg com os valores do novo processo que será executado, leva um overhead de tempo. (contexto de hw)

* + Dispacher: Armazena e recupera o contexto, atualiza as informações PCB, processo relativamente rápido (0,1ms).
  + Scheduler: Escalonador. Escolhe a próxima tarefa a ter acesso ao processador, parte mais demorada.

# Tabela de processos

Tabela com infos importantes de cada processo, incluindo o seu contador de programa, ponteiro de pilha, alocação de memória, estado dos arquivos abertos, informação sobre sua contabilidade e escalonamento, com o estado (em execução, pronto ou bloqueado).

Process Control Block (PCB): reside na memória principal e mantêm todas as informações sobre contexto de hardware, software e espaço de endereçamento de cada processo.

# Estados de um processo

* Executando: realmente usando a CPU naquele instante
* Pronto: executável, temporariamente parado para deixar outro processo ser executado.
* Bloqueado: incapaz de ser executado até que algum evento externo aconteça.

# Escalonador de processos

Responsável por realizar a troca de processos. O escalonador também é um processo. Diversas técnicas/algoritmos para realizar o escalonamento de processos. Escolhe qual é o próximo processo a ser executado. Nível mais baixo do SO.

* + Preemptivo: Processo pode perder uso da CPU, provoca uma interrupção forçada do processo para que outro possa usar.
  + Não Preemptivo: Permite que o processo que está sendo executado continue a execução. Para quando:
    - Termina de executar;
    - Solicita uma E/S;
    - Libera o processador voltando ao estado de pronto;

Throughput: números de processos terminados por unidade de tempo.

Turnaround: tempo transcorrido desde o momento em que o sw entra e o instante em que termina sua execução.

tempo de resposta: intervalo entre a chegada ao sistema e início de sua execução.

tempo de espera: soma dos períodos em que o processo estava no seu estado pronto.

# Algoritmos de escalonador

Para cada tipo de sistema (os 3 vistos anteriormente, batch, tempo real e compartilhado).

Sistemas Operacionais – Questões

1. Quais são as duas principais funções de um sistema operacional?
2. Qual é a diferença entre sistemas de compartilhamento de tempo e de multiprogramação?
3. Quais das instruções a seguir devem ser deixadas somente em modo núcleo?
   1. Desabilitar todas as interrupções.
   2. Ler o relógio da hora do dia.
   3. Configurar o relógio da hora do dia.
   4. Mudar o mapa de memória.