Departamento de Engenharia de Software, Campus I - Taguatinga, UniProjeção

Orientador: Prof. Rogério Oliveira.

Autor: Nádio Dib (201918579)

Atividade: resumo cap. 1 e 2, Sistemas Operacionais.

Resumo capítulo 1:

- **Sistemas operacionais** – são programas que tornam o *hardware* utilizável:
    - Oferece um ambiente de trabalho simples, seguro e eficiente como uma camada intermediadora entre o *hardware* e *firmware*;
    - Também pode ser visto como uma própria extensão do computador ou mesmo como um gerenciador de recursos físicos do computador;
    - É responsável por isolar o *hardware* das aplicações;
    - Deve adequar o uso do *hardware* sem o prejudicar;
    - Exigências:
        - Oferecer recursos do sistema de forma simples e transparente;
        - Gerenciar a utilização dos recursos existentes de forma eficiente;
        - Garantir a integridade e segurança dos dados armazenados, processos no sistema e recursos físicos;
        - Proporcionar uma interface adequada para melhor utilização pelo usuário.
- ***Hardware*** – são dispositivos que compõem o computador;
- ***Software*** – são programas que operam no computador, virtualmente;
- ***Firmware*** - são programas especiais armazenados de forma permanente no *hardware*, onde, geralmente, são compostos por componentes de memória não volátil (ROM, PROM ou EPROM). O *firmware* possui programas tipicamente escritos em linguagem de máquina ou *assembly*;
- **1643, Pascal –** máquina de calcular mecânica;
- **Final séc. XVIII, Jacquard** – máquina de tear para perfurar cartões;
- **1822, Babbage** – máquina de diferenças;
- **1870, Thomson** – máquina analógica para previsão de marés;
- **1890, Hollerith** – máquina de tabular;
- **1934, Zuse** – máquina eletromecânica programável;
- **1935, Atanasoft –** máquina eletrônica ABC;
- **1937, Neumann** – arquitetura genérica para o computador;
- **1939, Laboratório Bell** – primeira calculadora eletromecânica;
- Funcionamento do computador:
    - ***Fetch*** – um processador efetua um ciclo de busca por uma instrução na memória;
    - ***Decode*** – decodificação no ciclo seguinte (após *fetch*);
    - ***Execute*** – ação necessária para execução no último ciclo, determinada pelas instruções (após *decode*);

    - A repetição desses comportamentos ocorre indefinidamente, isto é, até que ocorra um erro grave ou uma instrução de parada tipo ***halt***.
- **1950** – transistores impulsionam a eletrônica e o avanço dos computadores;
- **Jobs** – nova implementação de dados organizados em conjuntos;
- **1957** – surgimento FORTRAN (IBM);
- **1959** – microcomputador PDP-1 (DEC);
- **1960:**
    - COBOL (Pentágono/EUA)
    - **Multiprogramação** – quando vários *jobs* estão na memória principal, simultaneamente, enquanto o processador é chaveado de um *job* para outro, fazendo-os avançarem enquanto os dispositivos periféricos são mantidos em uso quase constante.
- **Problemas no uso de processadores para fins científicos e comerciais:**
    - **CPU Bounded** – alta taxa de processamento e cálculo, porém E/S baixo (uso científico);
    - **I/O Bounded** – baixa taxa de processamento e cálculo, porém E/S alto (uso comercial).
- **Solução para problemas em uso científico / comercial** – divisão da memória em partes (partições), onde cada divisão poderia ser mantida em execução um *job*;
- **Sistemas multiprogramados** – são sistemas que permitiam o uso simultâneo do computador por diversos usuários através do pseudoparalelismo;
- **Pseudoparalelismo** – obtido com o chaveamento do processador entre vários processos (interfaces interativas);
- **1970 a 1980** – CPDs, *mainframe* e bureaus de processamento de dados;
- **1990** – *workstations*, computação pessoal portátil e interoperabilidade;
- **2000** – *grid computing*, *clusters* (agrupamentos) de computadores geograficamente dispersos pelo globo e interligados através da Internet ou por redes locais de alto desempenho;
- **Interatividade** - é um aspecto que considera se o usuário utiliza diretamente o sistema operacional, podendo receber as respostas deste, sem intermediação e por meio de intervalos de tempo razoável;
- **Tempo de resposta (*response time*)** – é uma medida de interatividade que representa o intervalo de tempo decorrido entre um pedido ou solicitação de processamento e a resposta produzida pelo sistema;
- **Tempo de reação (*reaction time*)** – considera o tempo decorrido entre a solicitação de uma ação e seu efetivo processamento;
- **Produtividade (*throughput*)** – expressa usualmente em tarefas completas por unidade de tempo, ou seja, é uma medida que relaciona o trabalho efetivamente produzido e o tempo utilizado para realização deste trabalho;
- O sistema operacional aparece como uma camada sobre o *hardware* e *firmware*, mas simultaneamente envoltória deste;
- Todos os elementos funcionais do computador são usualmente chamados de **plataforma** ou **ambiente computacional**;

- O conjunto de uma *interface*, gráfica ou não, para um sistema operacional é extremamente importante, pois determinam a criação de um ambiente operacional consistente:
    - Determinação de um ambiente de trabalho equivalente para os usuários;
    - Criação de um ambiente de desenvolvimento semelhante;
    - Redução das necessidades de treinamento e aceleração do processo de familiarização e aprendizado no novo ambiente.

Resumo capítulo 2:

- **Processo computacional** – é uma atividade que ocorre no meio computacional, usualmente possuindo um objetivo definido, tendo duração finita e utilizando uma quantidade limitada de recursos computacionais:
    - O termo processo (*process*) é muitas vezes substituído pelo termo tarefa (*task*).
- **Subprocessos** – são processos computacionais que podem ser divididos ou decompostos em processos componentes mais simples, o que permite um detalhamento da realização de sua tarefa ou do seu modo de operação;
- **Os processos tipicamente criam processos:**
    - **Processo-pai (*parente process*)** – é o processo criador;
    - **Processo-filho (*child process*)** – é um processo criado por um processo-pai.
- **Processos sequenciais** – são executados um de cada vez;
- **Processos paralelos** – são executados ao mesmo tempo:
    - **Independentes –** são processos paralelos que quando utilizam recursos completamente distintos, não se relacionam em disputas com outros processos;
    - **Concorrentes –** são processos paralelos que quando pretendem utilizar um mesmo recurso, dependem de uma ação do sistema operacional para definir a ordem de utilização;
    - **Cooperantes –** é quando dois ou mais processos utilizam em conjunto um mesmo recurso para completarem uma dada tarefa.
- **Estados dos processos:**
    - Pronto (*ready*);
    - Execução (*running*);
    - Bloqueado (*blocked*).
- **Situações de transição para modificação do estado do processo:**
    - Despachar (*dispatch*);
    - Esgotamento (*time run out*);
    - Bloqueio (*block*);
    - Reativar (*awake*).
- **Outras correspondências de transições:**
    - Criação (*create*) – utiliza-se o **PID** ou número de identificação do processo;
    - Finalização (*terminate*).
- **Escalonador (*scheduler*) –** coordena a utilização do processador por parte dos processos;

- **Tabela de processos –** organiza as informações relativas aos processos;
- **PCB (*Process Control Block* ou *Process Descriptor*) –** é uma estrutura de dados que mantém a representação de um processo para o sistema operacional;
- **Operação entre processos:**
    - Criação (*create*);
    - Destruição (*destroy*);
    - Suspensão (*suspend* ou *wait*);
    - Retomada (*resume*);
    - Troca de prioridade;
    - Bloqueio (*block*);
    - Ativação (*activate*);
    - Execução (*execute* ou *dispatch*);
    - Comunicação inter-processo;
    - Identificação do processo (PID);
    - Inserção do processo na lista de processos conhecidos do sistema;
    - Determinação da prioridade inicial do processo;
    - Criação do PCB;
    - Alocação inicial dos recursos necessários.
- Todas as operações que envolvem os processos são controladas por uma parte do sistema operacional denominada **núcleo** (*core* ou *kernel*), onde é responsável pelo gerenciamento das interrupções;
- Rotinas que devem conter no núcleo do sistema operacional:
    - Gerenciamento de interrupções;
    - Manipulação dos processos;
    - Manipulação dos PCBs;
    - Troca de estados dos processos;
    - Intercomunicação de processos (IPC);
    - Sincronização de processos;
    - Gerenciamento de memória;
    - Gerenciamento de dispositivos de E/S;
    - Suporte a um ou mais sistemas de arquivos;
    - Suporte às funções de administração do sistema.
- Quando afirmamos que existirão dois ou mais processos em execução paralela estamos admitindo a possibilidade de que alguns destes processos solicitem a utilização de um mesmo recurso simultaneamente;
- **Região crítica (*critical section*) -** quando um dado recurso computacional só pode ser utilizado por um único processo de cada vez;
- Uma região crítica pode ser uma rotina de software especial, um dispositivo de *hardware* ou mesmo uma rotina de acesso para um dispositivo do *hardware*;
- **Exclusão mútua** – quando um processo qualquer utiliza a região crítica, todos os demais ficam impossibilitados de acessar o recurso compartilhado dessa região enquanto o processo estiver alterando-a ou consultando o registro;
- Os processos que desejam utilizar uma região crítica são **processos paralelos concorrentes**;
- **Acesso simultâneo -** é quando um ou mais processos estão acessando a mesma região crítica aonde está alocado o recurso compartilhado;

- **Protocolo de acesso** – é responsável por determinar os critérios de utilização dos recursos, resolver situações de competição pelos recursos, bem como a organização de uma eventual lista de espera em caso de disputas concorrentes entre processos paralelos;
- **Código reentrante** ou **código público (*public code*) –** utilizados por uma quantidade indefinida de processos onde compartilham do mesmo recurso, aumentando a eficiência do sistema;
- **Protocolo de acesso (*access protocol*):**
    - Composto por uma rotina de entrada e outra de saída;
    - Problemas encontrados:
        - **Prioridade estática -** ocorre quando um processo de prioridade mais baixa não consegue utilizar a região crítica e acessar o recurso, dado que sempre existirá processos de maior prioridade;
        - **Bloqueio simultâneo -** vários processos que estão em disputa pelo uso do recurso são bloqueados, de modo que o recurso fique inutilizado;
        - **Adiamento infinito -** quando um processo é bloqueado indefinidamente de utilizar o recurso.
- **TST** ou **TSL (*Test and Set*** ou ***Test and Set Lock*)**;
- Requisitos desejáveis para um protocolo de acesso eficiente (postulado de Dijkstra):
    - A solução não deve impor uma prioridade estática entre os processos que desejam acessar a região crítica;
    - A velocidade de execução dos processos paralelos não é nula;
    - Se um processo é bloqueado fora da região crítica, isto não deve impedir que outros processos acessem a mesma região;
    - É inaceitável situações de bloqueio mútuo ou acesso simultâneo de processo em uma região crítica.
- **Solução de Dekker;**
- **Solução de Peterson;**
- ***Deadlocks*** – evento que jamais vai ocorrer:
    - A maioria dos problemas estão relacionados com recursos dedicados, ou seja, são recursos que podem ser utilizados por um processo somente cada vez;
    - Condições de ocorrência:
        - Os processos exigem controle exclusivo dos recursos que solicitam;
        - Os processos mantêm alocados recursos enquanto solicitam novos recursos;
        - Os recursos não podem ser retirados dos processos que os mantém alocados enquanto estes processos não são finalizados;
        - Forma-se uma cadeia circular de processos, onde cada um solicita um recurso alocado pelo próximo, resultando em um processo em cadeia ou circular.
    - Soluções:
        - Ignorar o problema;

        - Detecção e recuperação;
        - Prevenção dinâmica;
        - Prevenção estrutural.
    - Solução através da **recuperação**:
        - Preempção;
        - Operações de *rollback*;
        - Eliminação processual.
    - Solução através da **prevenção**:
        - Um processo só pode solicitar um recurso após liberar o mesmo recurso;
        - Um processo que é negado o pedido de acesso ao recurso da região crítica deve liberar o pedido;
        - Se a solicitação de recursos ocorrer em ordem linear, ou seja, de forma sequencial, a espera circular não se formará.
- **Algoritmo do Banqueiro, por E.W. Dijkstra em 1965 (*Banker's Algorithm*)** – efetua-se um mapeamento dos recursos e processos de forma a considerar que cada pedido de uso de um recurso esteja em um estado seguro. Se o estado seguinte for seguro, então o pedido é concedido, caso contrário, a solicitação é abortada e adiada até que um estado seguro para o processo seja obtido;
- IPC (*Inter Process Communication*) ocorre quando os processos compartilham dados entre si;
    - Para existir, é necessário algum mecanismo bem estruturado dentro do sistema operacional;
    - A comunicação se dá através da utilização de recursos comuns nos processos envolvidos;
    - As interrupções são reservadas para a administração do sistema em si.
- ***Buffers*** – é uma região crítica acessada por processos paralelos concorrentes, além disso é uma área de dados de tamanho fixo que se comporta como um reservatório temporário ou volátil:
    - **Processo produto** – coloca informações no *buffer*;
    - **Processo consumidor** – extrai informações do *buffer*;
    - Tanto processos produto quanto consumidor são sequenciais e dentre operações em paralelo é criado uma situação de concorrência por informações específicas do *buffer*;
    - Tipos de problemas:
        - ***Buffer* cheio** – o processo produtor é incapaz de colocar novas informações;
        - ***Buffer* vazio** – o processo consumidor é incapaz de extrair informações.
    - Soluções:
        - Modo adormecido (*sleep*);
        - Modo acordar (*wake up*).
        - As soluções podem ser parciais, ou seja, caso um processo são esteja em modo adormecido ou de suspensão e um sinal de acordar for sinalizado, os processos permanecerão em estado indefinido;

        - Para certos procedimentos de sinalização para o modo acordar é necessário inicializar uma *flag* para notificar o estado atual do processo enquanto operar no *buffer*.
- **Semáforos** – garantem que um dado recurso seja compartilhado de forma sequencial entre processos concorrentes, evitando problemas na região crítica:
    - **P (*down*);**
    - **V (*up*).**
- **Memória compartilhada** - é uma região da memória reservada para uso comum dos processos envolvidos no IPC;
- Outros mecanismos IPC:
    - Contadores de eventos (*event counters*);
    - **Monitor** - é uma coleção de procedimentos, variáveis e estruturas agrupadas num módulo ou pacote especial.
- ***Threads*** – são fluxos independentes de execução, onde pertencem a um mesmo processo, ou seja, requerem menos recursos de controle por parte do sistema operacional;
- Modelos **multithreading**:
    - Modelo **"n para 1"** – é aonde vários *threads* do usuário são associados a um único processo suportado pelo sistema operacional;
    - Modelo **"1 para 1"** – é quando cada *thread* do usuário se associa a outra *thread* nativa do sistema operacional;
    - Modelo **"n para m"** – é quando um conjunto de *threads* de usuário associam-se a outro conjunto de *threads* nativas do sistema operacional.