

SO-T 2015 (3rd)

A

1

a) Na perspectiva do utilizador o SO é o programa base que vai estabelecer uma interface entre o user e o hardware da sua máquina. Vai providenciar um nível de abstracção que permitirá ao user executar funções e manipulações permitidas pelo hardware. Sem que este tenha de se preocupar com os procedimentos e detalhes de como o hardware trabalha (device drivers).

b) Doutro ponto de vista (bottom's up), o SO está encarregue de gerir de forma eficiente os recursos disponibilizados pelo CPU e restante hardware.

c) O ambiente de interacção com o utilizador (gráfico ou por linha de comandos) permite ao utilizador realizar manipulações e comandos simples do seu sistema de forma simples e eficiente.

2 Um programa é uma entidade passiva que contém trechos de código a ser executado. Um processo é, basicamente, um programa ativo. Ao executar um programa é criado um processo que o inicializa. Este processo inicial pode depois no decorrer da execução do código dar origem a processos filhos e estes podem por sua vez ter também filhos, e por ai em diante. Cada programa pode portanto dar origem a vários processos, sendo designado um "anão de processos". Cada processo ~~possui~~ contém um PCB que possui info. relativamente a este, pode passar em uma ou várias threads, comunicar com outras por mensagens ou memória partilhada e concorrer por recursos (so 1 Proc. pode conter num CPU de cada vez).

3

a) Em preemptive scheduling, o SO pode fazer com que um processo deixe de ocupar o CPU, mesmo que este ainda tenha terminado, recolocando-o no estado ready.

Nos escalonadores non preemptive um processo apenas sai do CPU quando transita para o estado Terminated (quando termina)

c) Em prioridade estática, como é o caso do FCFS, cujos sen degrada a ordem / prioridade de execução dos processos, esta não pode ser alterada. Em algoritmos como o SJF, porém, a prioridade de um processo executar pode diminuir caso chegue um processo com menor CPU Burst, ou seja, a prioridade individualmente dada aos processos pode alterar, dinamicamente.

B

1. Região Crítica → Métodos `retrieveValFromCommBuff(...)`
`insertValFromCommBuff(...)` pois
São métodos que não manipulam
dados em memória partilhada

(OU, métodos `getVal` e `setVal`, not
sure -_-('`)-/-)

Região Partilhada → ?