Sistemas operacionais: Conceitos e Mecanismos

INSTITUTO FEDERAL
Santa Catarina

Capítulo 5: Implementação de tarefas

Integrantes do grupo: Jefferson Botitano e Leonardo Ludvig
Professor: Arliones Stevert Hoeller Junior
Disciplina: Sistemas operacionais

Campus São José

Contexto e Processos

INSTITUTO FEDERAL
Santa Catarina

- **Contexto:** é o estado de uma tarefa em um determinado tempo, parte da relevância do contexto da tarefa está ligada ao estado interno do processador durante sua execução, como o valor do contador de programa (PC), do apontador de pilha (SP) e outra parte está contendo informações sobre os recursos usados pela tarefa.
- **Troca de contexto:** ação de suspender uma tarefa e reativar outra, a recuperação de contexto e atualização de informações contidas no TCB(Task control block) de cada tarefa são aspectos mecânicos que são formados por um conjunto de rotinas chamadas despachante ou executivo (dispatcher) e a escolha da próxima tarefa a ser executada no processador é feita de forma que vários fatores implicam nesta decisão como prioridades, tempos de vida e tempos de processamento restante de cada tarefa que são decididas pelo escalonador(scheduler).
- **Processos:** contém uma tarefa e seus recursos, sendo mais visado como unidade de contexto, que pode ser o equivalente a um contêiner de recursos que são utilizados por uma ou mais tarefas para sua execução e são isolados entre si pelo mecanismo de proteção providos pelo hardware para que não ocorra acessos de recursos errados entre as tarefas.
- **Gestão de processos:** operações de criação e destruição de processos são disponibilizadas as aplicações por meio de chamadas de sistema e que varia para cada sistema operacional tendo suas próprias chamadas para a gestão de processos, ou seja a sintaxe de como gerir processos varia para cada sistema operacional.
- **Hierarquia de processos:** os processos novos são considerados filhos e os que criaram são considerado pai e em relação a isto varia de acordo a casa sistema operacional.

Threads

INSTITUTO FEDERAL
Santa Catarina

As threads surgiram da necessidade de executar mais de uma tarefa por processo em um sistema operacional, uma thread é um fluxo de execução independente contendo um código e um pequeno contexto local, sendo possível um processo possuir várias threads. Para gerenciar essas threads contidas em um processo definidas no espaço de usuário foram criados alguns modelos.

- **O modelo N:1:** Nesse modelo cada thread dentro de um processo opera por vez, visto que para o núcleo existe apenas uma thread de núcleo, não sendo possível a execução de threads em paralelo no mesmo processo.
- **O modelo 1:1:** Para contornar esse problema do paralelismo o modelo 1:1 cria uma thread de núcleo para cada thread de usuário no processo, também resolvendo o problema de uma thread bloquear as outras thread do mesmo processo ao ser suspensa ou interrompida, porém a escalabilidade se torna muito menor.
- **O modelo N:M:** Um híbrido entre os dois modelos anteriores possui as qualidades dos dois, mas com as desvantagens de uma alta complexidade de implementação e alto custo de gerência de threads de núcleo.

Uso de processos versus threads

INSTITUTO FEDERAL
Santa Catarina

Podemos implementar tarefas através das duas formas apresentadas, com os processos e as threads, com cada um tendo as suas vantagens, sendo o método de processos muito mais robusto pois ao executar apenas uma tarefa por vez os erros são restritos ao seu espaço de memória, podendo ser executadas com permissões e usuários distintos aumentando a segurança do sistema, porém trazendo como desvantagem a dificuldade de compartilhamento de dados entre aplicações. Já nos sistemas com threads é possível executar todas as tarefas em um único processo, usando um. a thread por tarefa, como já mostrado anteriormente. Nos sistemas atuais é adotada uma abordagem híbrida que busca somar as qualidades das duas implementações aliando a robustez proporcionada pela divisão de memória entre processos e o alto desempenho das threads.

Explique o que é, para que serve e o que contém um TCB- Task Control Block.

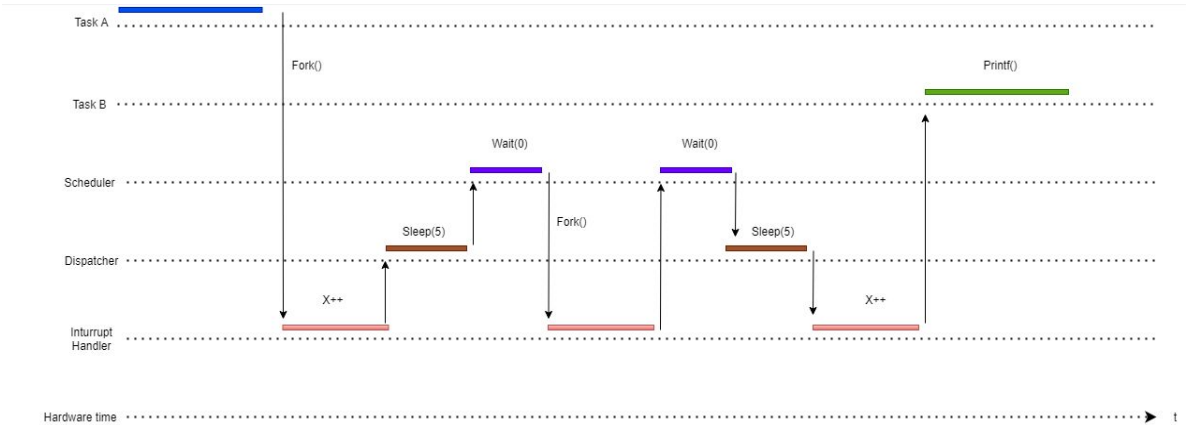
INSTITUTO FEDERAL
Santa Catarina

Um descritor é representar dados organizados no núcleo de forma que cada tarefa presente está sendo representado no sistema operacional, ou seja , esse tipo de organização de dados contém as informações de importância relativa ao seu tipo de contexto e os demais dados e outro tipos de dados são gerenciados por prioridades, estado , entre outros. Um TCB é composto por informações do tipo identificador da tarefa, estado da tarefa, contexto do processador,lista de áreas de memória usados pela tarefa,listas de arquivos abertos, conexões de rede e outros recursos que são utilizados quando requisitados pela tarefa, gerência e contabilização.

Desenhe o diagrama de tempo da execução do código a seguir,informe qual a saída do programa na tela (com os valores de x) e calcule a duração aproximada de sua execução

```c
int main()
{
   int x = 0 ;

   fork () ;
   x++ ;
   sleep (5) ;
   wait (0) ;
   fork () ;
   wait (0) ;
   sleep (5) ;
   x++ ;
   printf ("Valor de x: %d\n", x) ;
}
```




```
~/D/SOP  ./cap5ex3                    sáb 05 jun 2021 20:36:22
Valor de x: 2
Valor de x: 2
Valor de x: 2
Valor de x: 2
~/D/SOP                          25s 〈 sáb 05 jun 2021 20:36:49
```

Indique quantas letras “X” serão impressas na tela pelo programa abaixo quando for executado com a seguinte linha de comando: a.out 4 3 2 1
O comando a.out resulta da compilação do programa a seguir:

```c
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
   pid_t pid[10];
   int i;

   int N = atoi(argv[argc-2]);

   for (i=0; i<N; i++)
      pid[i] = fork();
   if (pid[0] != 0 && pid[N-1] != 0)
      pid[N] = fork();
   printf("X");
   return 0;
}
```

Código entra no loop duas vezes e pai irá criar 2 filhos,o primeiro filho irá executar mais uma vez o loop sendo que o mesmo foi criado com sua variável i igual a zero resultando em 4 filhos, e caindo no condicional que diz que se não for um filho execute a linha de baixo, resultando em 5 processos e em seguida realizando printf de X, resultando 5 impressões de X .

```
~/D/SOP ./cap5ex4 4 3 2 1
XXXXX⏎
~/D/SOP
```

O que são threads e para que servem?

INSTITUTO FEDERAL
Santa Catarina

Threads são fluxos de execuções independentes, elas servem para executar o código de uma aplicação dentro de um processo, podendo existir diversas threads em um único processo executando de forma paralela ou não dependendo do modelo de thread adotado pelo sistema operacional. Com as threads surgiu a possibilidade de executar mais de uma tarefa por processador, o que possibilitou a execução de aplicações não possíveis antes.

Quais as principais vantagens e desvantagens de threads em relação a processos

INSTITUTO FEDERAL
Santa Catarina

As threads tem como principais vantagens em relação aos processos a capacidade de executar mais de uma tarefa no processador, utilizando dos mesmos endereçamentos e recursos como arquivos abertos e conexões de rede, trazendo uma maior agilidade, escalabilidade e velocidade para o sistema operacional, porém possui a desvantagem de uma thread ao apresentar um erro pode acabar alastrando para as outras thread do mesmo processo comprometendo a robustez da aplicação inteira.

Associe as afirmações a seguir aos seguintes modelos de threads: a)many-to-one (N:1); b) one-to-one (1:1); c) many-to-many (N:M):

INSTITUTO FEDERAL
Santa Catarina

(a) Tem a implementação mais simples, leve e eficiente. **A**

(b) Multiplexa os threads de usuário em um pool de threads de núcleo. **B**

(c) Pode impor uma carga muito pesada ao núcleo. **B**

(d) Não permite explorar a presença de várias CPUs pelo mesmo processo. **A**

(e) Permite uma maior concorrência sem impor muita carga ao núcleo. **C**

(f) Geralmente implementado por bibliotecas. **A**

(g) É o modelo implementado no Windows NT e seus sucessores. **B**

(h) Se um thread bloquear, todos os demais têm de esperar por ele. **A**

(i) Cada thread no nível do usuário tem sua correspondente dentro do núcleo. **B**

(j) É o modelo com implementação mais complexa. **C**