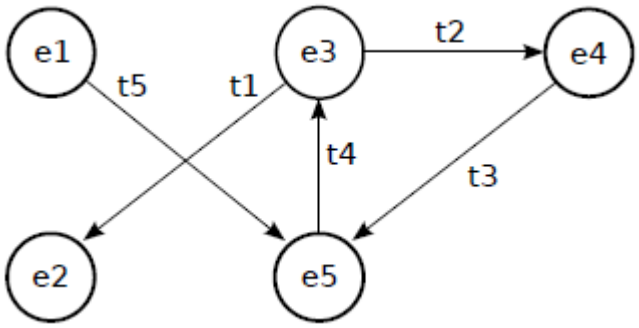
SO 2020.2 – Ecomp POLI UPE

Fonte: Maziero, 2019.

1. O que significa *time sharing* e qual a sua importância em um sistema operacional?

2. Como e com base em que critérios é escolhida a duração de um quantum de processamento?

3. Considerando o diagrama de estados dos processos apresentado na figura a seguir, complete o diagrama com a transição de estado que está faltando (t_6) e apresente o significado de cada um dos estados e transições.



4. Indique se cada uma das transições de estado de tarefas a seguir definidas é possível ou não. Se a transição for possível, dê um exemplo de situação na qual ela ocorre (N: Nova, P: pronta, E: executando, S: suspensa, T: terminada).

E → P
E → S
S → E
P → N
P → S
S → T
E → T
N → S

5. Relacione as afirmações abaixo aos respectivos estados no ciclo de vida das tarefas (N: Nova, P: Pronta, E: Executando, S: Suspensa, T: Terminada):

[ ] O código da tarefa está sendo carregado.
[ ] A tarefas são ordenadas por prioridades.
[ ] A tarefa sai deste estado ao solicitar uma operação de entrada/saída.
[ ] Os recursos usados pela tarefa são devolvidos ao sistema.
[ ] A tarefa vai a este estado ao terminar seu quantum.
[ ] A tarefa só precisa do processador para poder executar.
[ ] O acesso a um semáforo em uso pode levar a tarefa a este estado.
[ ] A tarefa pode criar novas tarefas.
[ ] Há uma tarefa neste estado para cada processador do sistema.
[ ] A tarefa aguarda a ocorrência de um evento externo.

6. Explique o que é, para que serve e o que contém um T*CB - Task Control Block.*

7. O que são *threads* e para que servem?

8. Quais as principais vantagens e desvantagens de *threads* em relação a processos?

9. Forneça dois exemplos de problemas cuja implementação *multi-thread* não tem desempenho melhor que a respectiva implementação sequencial.

10. 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):

(a) Tem a implementação mais simples, leve e eficiente.
(b) Multiplexa os *threads* de usuário em um pool de *threads* de núcleo.
(c) Pode impor uma carga muito pesada ao núcleo.
(d) Não permite explorar a presença de várias CPUs pelo mesmo processo.
(e) Permite uma maior concorrência sem impor muita carga ao núcleo.
(f) Geralmente implementado por bibliotecas.
(g) É o modelo implementado no Windows NT e seus sucessores.
(h) Se um *thread* bloquear, todos os demais têm de esperar por ele.
(i) Cada *thread* no nível do usuário tem sua correspondente dentro do núcleo.
(j) É o modelo com implementação mais complexa.