Python e Multi-threading

1. DEFININDO MULTITHREADING

Segundo o que escreve Baum; para a Gestão de Projetos, o conceito de tarefas é de processos em um nível superior de detalhamento. E, processos seriam atividades que agregam valor, recebem entradas transformando-as em resultados.

Conforme De Lima, "processo é um grupo de atividades realizadas numa sequência lógica com o objetivo de produzir um bem, ou serviço que tem valor para um grupo específico de clientes" (apud Hammer e Champy (1997)).

Para Pereira, "Uma *thread* é um conjunto de tarefas existentes em um ou mais programas, executadas ao mesmo tempo pelo processador". E, *Multithread*, "[...] é a capacidade que o sistema operacional possui de executar várias *threads* simultaneamente sem que uma interfira na outra".

Como já expresso, a thread lida com tarefas cuja a definição esclarece que o detalhamento descreve o que são processos. Logo, tarefas que se deseja desempenhar de forma concomitante e independente cujos os processos estão separados e não obstruem o resultado são, por conseguinte, o princípio do Multithreading.

2. UTILIDADES DE MULTITHREADING

Conforme Pereira, “Por exemplo, você não precisa para de ouvir música enquanto utiliza um editor de texto, muito fechar uma janela de seu navegador para imprimir uma imagem”. De fato, atualmente não precisamos parar de ouvir música para utilizar um editor texto. Continua Pereira, “Agora pense em diversas tantas tarefas dentro de um único processo (como a emissão de sons e imagens ao mesmo tempo em um jogo) e você tem threads.

A implementação de *multithreads* pode fornecer um melhor desempenho, segundo Maia, a aplicações que trabalham com matrizes, equações, ordenações, pesquisas e processamento de imagens dando-as ganhos significativos com o aumento de paralelismo é permitido.

3. MODELOS

Assim, define-se:

> "No modelo de grupo de trabalho (*workgroup*, *peer-to-peer* ou *workcrew*), todos os threads trabalham de forma concorrente para a solução cooperativa do problema, onde cada executa mais de uma tarefa bem especificada" (Maia)

Outro tipo:

> "No modelo mestre-escravo (*master-slave*) ou chefe trabalhador (*boss-worker*) existe um *thread* principal (mestre), responsável por criar, eliminar e coordenar os *threads* que efetivamente executarão as tarefas (escravos)". (Maia)

E um último:

> "O modelo de *pipeline* deve ser empregado quando um problema pode ser dividido em vários *threads*, por onde os dados ao sair de um *thread* podem ser processados pelo *thread* seguinte, como em uma linha de produção". (Maia)

4. REFERÊNCIAS

Baum, E. "**Conceito básicos de atividades, tarefas, processos e macroprocessos**". Disponível em <https://www.sabesim.com.br/conceitos-atividades-tarefas-processos-macroprocesso/>. Acessado em 07 de junho de 2019.

De Lima, F. U. "**Processos Organizacionais"**. Disponível em <http://www2.unifap.br/furtado/files/2017/04/Processos_Organizacionais_1.pdf> Acessado em 8 de junho de 2019.

Maia, L. P.. <http://www.training.com.br/lpmaia/multithread.pdf> Acessado em 07 de junho de 2019.

Pereira, A. P.. "**O que são multi e hyper-threading?"**. Disponível em: <https://www.tecmundo.com.br/aumentar-desempenho/2841-o-que-sao-multi-e-hyper-threading-.htm> Acessado em 7 de junho de 2019.