



Thread

Professor Danilo

Introdução

- **Até o final da década de 1970, sistemas operacionais**, como Tops-10 (DEC), MVS (IBM) e Unix (Bell Labs), suportavam apenas processos com um único thread (**monothread**), ou seja, um **processo com apenas um único programa fazendo parte do seu contexto**.
- Em **1979**, durante o desenvolvimento do sistema operacional Toth, foi introduzido o conceito de processos **lightweight** (peso leve), onde o **espaço de endereçamento** de um processo era **compartilhado por vários programas**.
- **Somente em meados de 1980**, com o desenvolvimento do sistema operacional Mach, na Universidade de Carnegie Mellon, **ficou clara a separação entre o conceito de processo e thread**.

Introdução

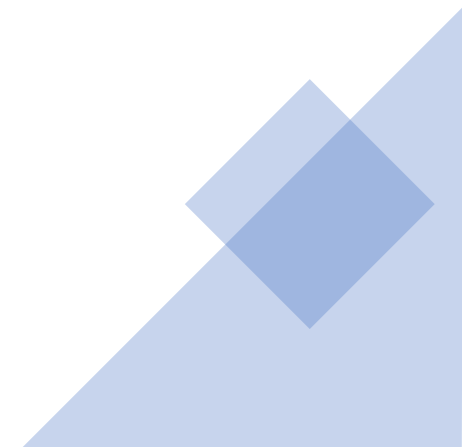
- A partir do conceito de **múltiplos threads (multithread)** é possível projetar e implementar aplicações concorrentes de forma eficiente, pois **um processo pode ter partes diferentes do seu código sendo executadas concorrentemente, com um menor overhead (sobrecarga) do que utilizando múltiplos processos.**
- **Como os threads de um mesmo processo compartilham o mesmo espaço de endereçamento, a comunicação entre threads não envolve** mecanismos lentos de intercomunicação entre processos, aumentando, conseqüentemente, **o desempenho da aplicação.**

Introdução

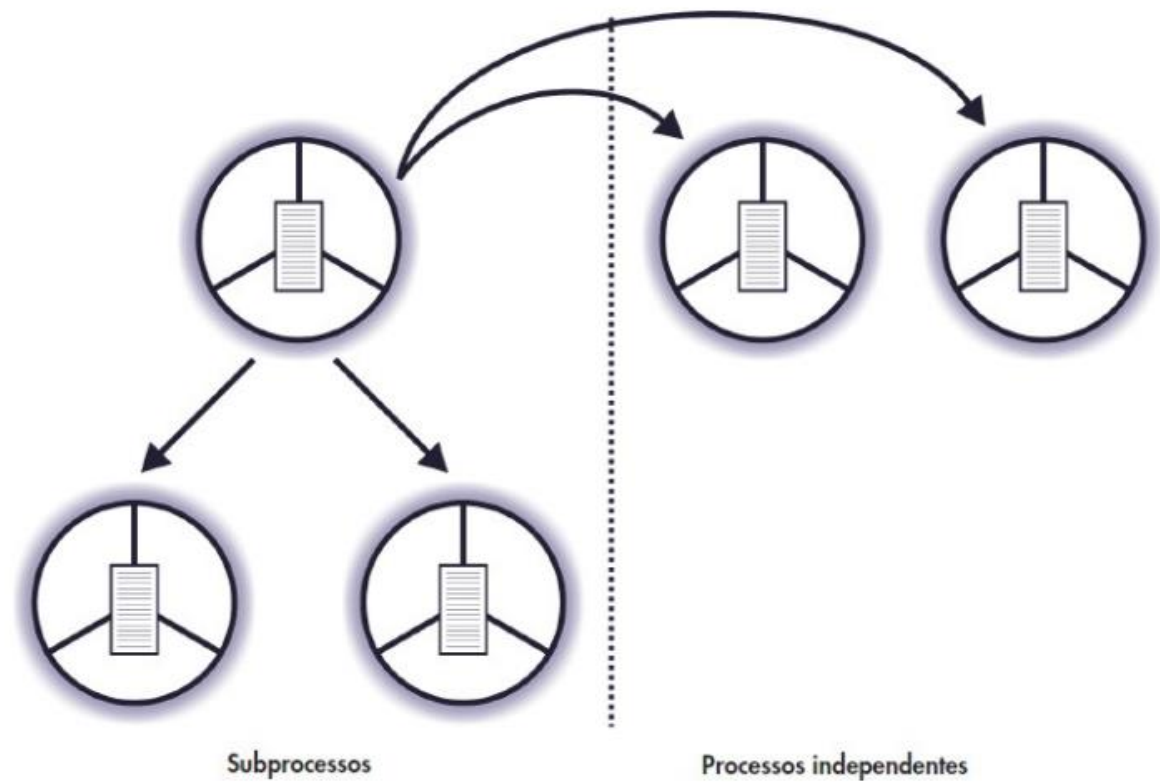
- **O desenvolvimento de programas que exploram os benefícios da programação multithread não é simples.** A presença do paralelismo introduz um novo conjunto de problemas, como a comunicação e sincronização de threads. Existem diferentes modelos para a implementação de threads em um sistema operacional, onde desempenho, flexibilidade e custo devem ser avaliados.
- Atualmente, **o conceito de multithread pode ser encontrado em diversos sistemas operacionais, como no Sun Solaris e MS Windows.** A utilização comercial de sistemas operacionais multithread é crescente em função do aumento de popularidade dos sistemas com múltiplos processadores, do modelo cliente servidor e dos sistemas distribuídos.



Ambiente Monothread

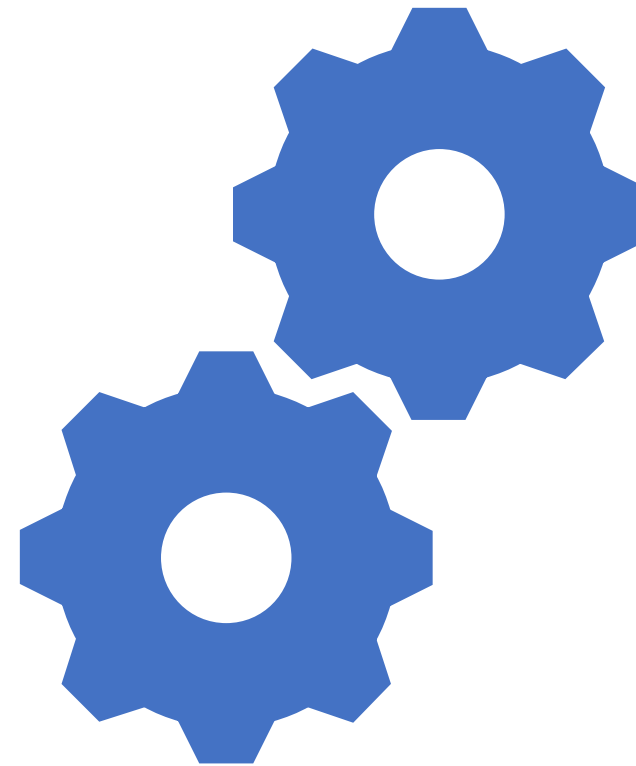
- Um programa é uma sequência de instruções, composta por desvios, repetições e chamadas a procedimentos e funções.
 - **Em um ambiente monothread, um processo suporta apenas um programa no seu espaço de endereçamento.**
 - **Aplicações concorrentes são implementadas apenas com o uso de múltiplos processos independentes ou subprocessos.**
- 

Ambiente Monothread



Ambiente Monothread

- A utilização de processos independentes e subprocessos permite dividir uma aplicação em partes que podem trabalhar de forma concorrente.
- **Um exemplo do uso de concorrência pode ser encontrado nas aplicações com interface gráfica, como em um software de gerenciamento de e-mails.** Neste ambiente, um usuário pode estar lendo suas mensagens antigas, ao mesmo tempo que pode estar enviando e-mails e recebendo novas mensagens. Com o uso de múltiplos processos, cada funcionalidade do software implicaria a criação de um novo processo para atendê-la, aumentando o desempenho da aplicação.



Ambiente Monothread

- O **problema** neste tipo de implementação é que o **uso de processos** no desenvolvimento de **aplicações concorrentes demanda consumo de diversos recursos do sistema**.
- Sempre que um novo processo é criado, o sistema deve alocar recursos para cada processo, consumindo tempo de processador neste trabalho. No caso do término do processo, o sistema dispensa tempo para desalocar recursos previamente alocados.

Ambiente Monothread

- **Outro problema** a ser considerado é quanto ao **compartilhamento do espaço de endereçamento**. Como cada processo possui seu próprio espaço de endereçamento, **a comunicação entre processos torna-se difícil e lenta**.
- Além disso, o compartilhamento de recursos comuns aos processos concorrentes, como memória e arquivos abertos, não é simples.
- Existem três processos monothreads, cada um com seu próprio contexto de hardware, contexto de software e espaço de endereçamento.

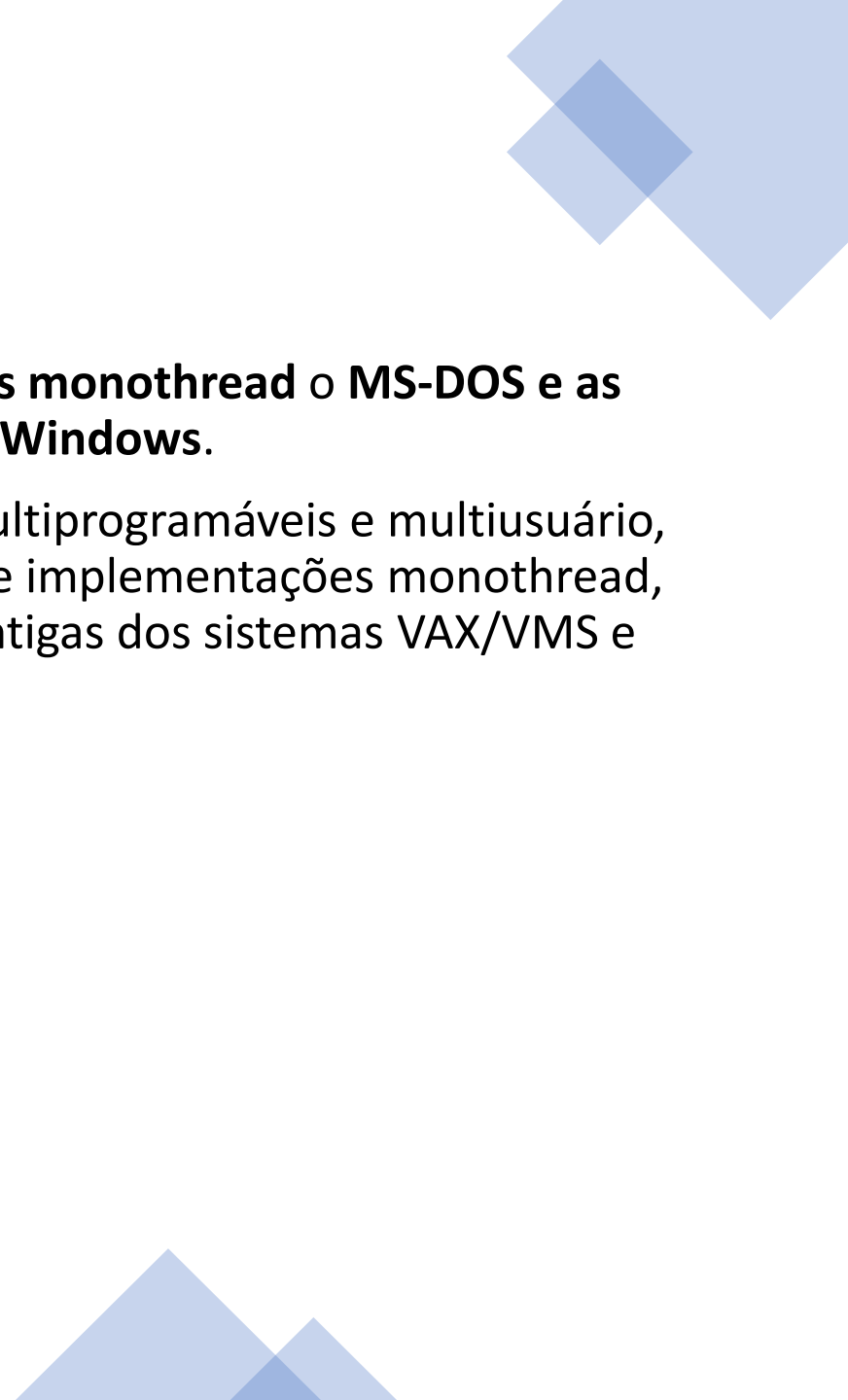
Ambiente Monothread



Fig. 6.2 Ambiente monothread.

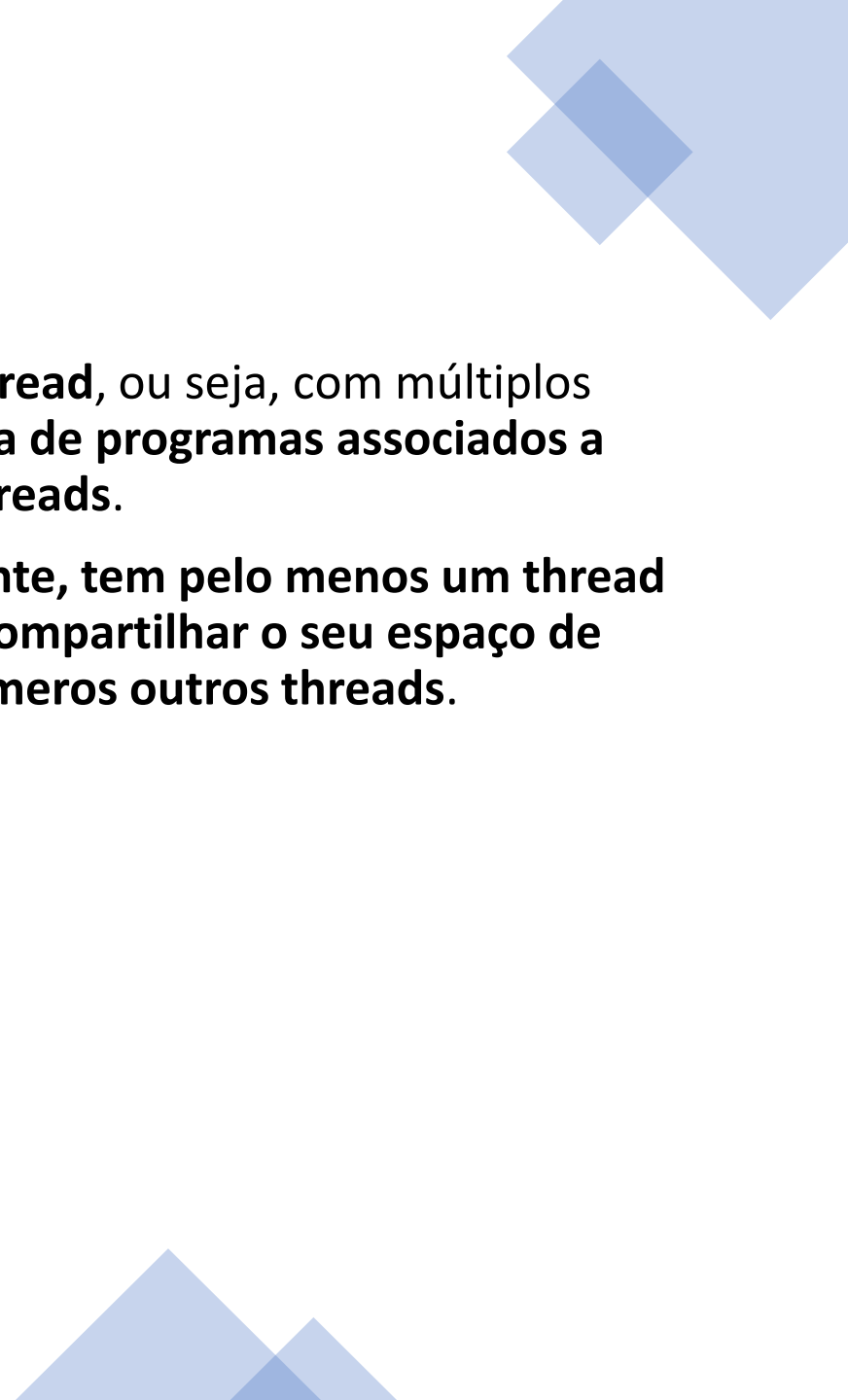


Ambiente Monothread

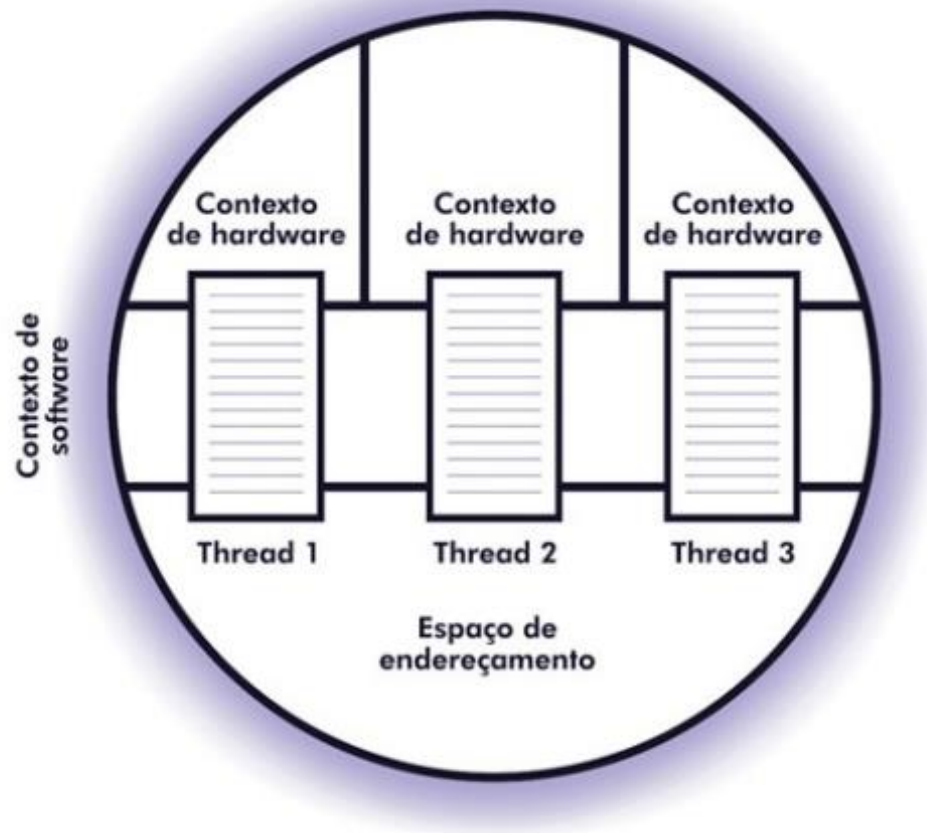
- São **exemplos de sistemas monothread** o **MS-DOS** e as **primeiras versões do MS Windows**.
 - Mesmo em ambientes multiprogramáveis e multiusuário, encontramos exemplos de implementações monothread, como nas versões mais antigas dos sistemas VAX/VMS e Unix.
- 



Ambiente Multithread

- **Em um ambiente multithread, ou seja, com múltiplos threads, não existe a ideia de programas associados a processos, mas, sim, a threads.**
 - **O processo, neste ambiente, tem pelo menos um thread de execução, mas pode compartilhar o seu espaço de endereçamento com inúmeros outros threads.**
- 

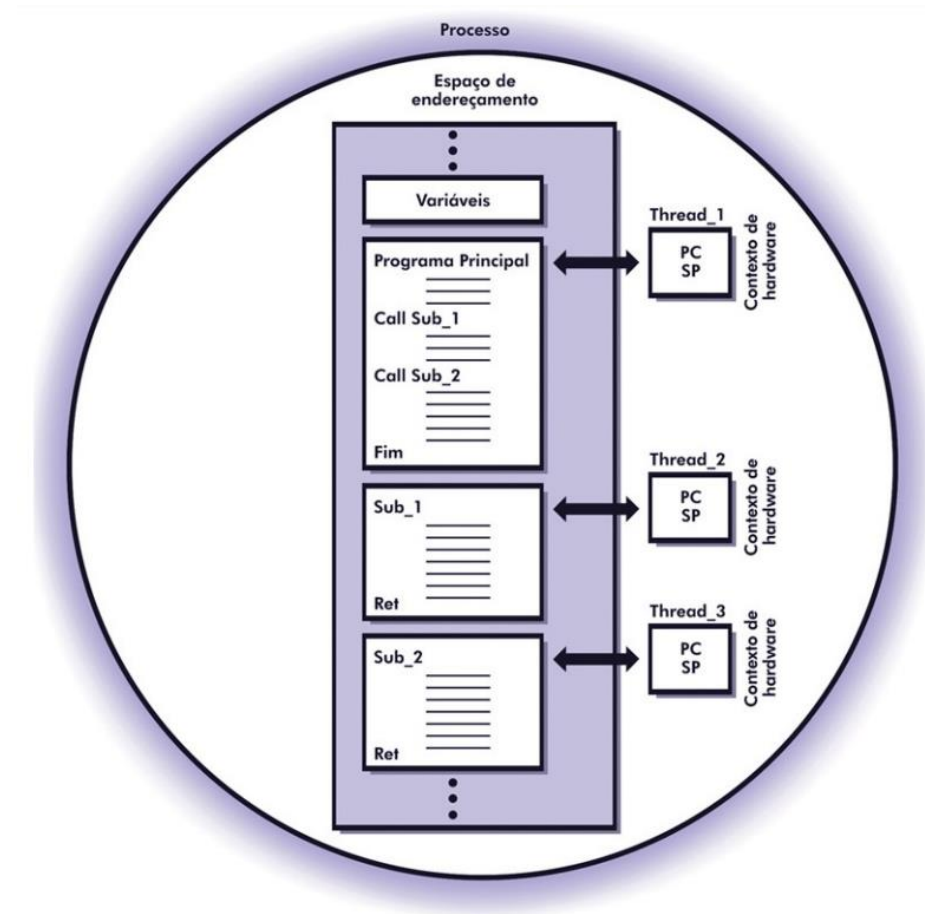
Ambiente Multithread



Ambiente Multithread

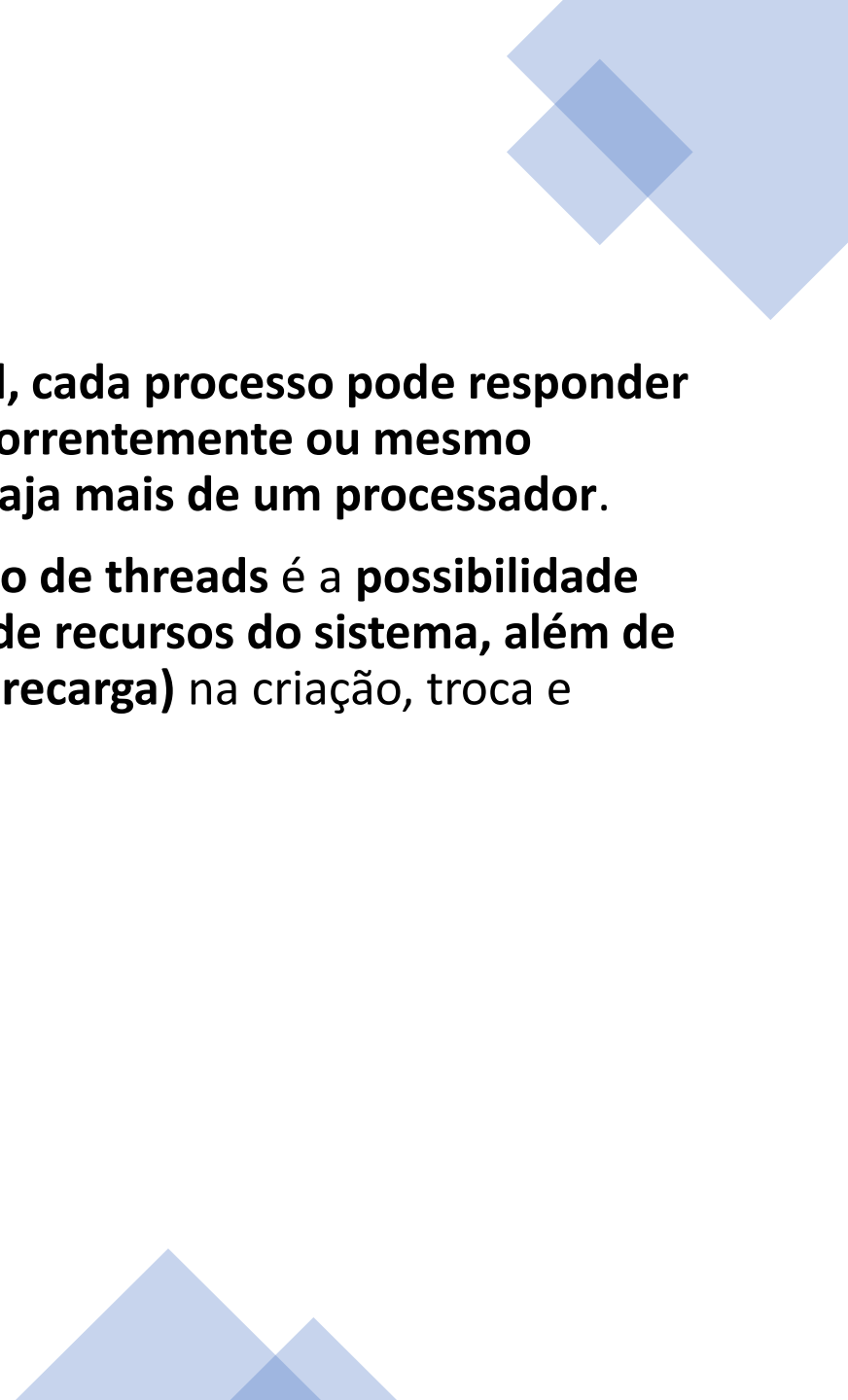
- De forma simplificada, um thread pode ser definido como uma sub-rotina de um programa que pode ser executada de forma assíncrona, ou seja, executada concorrentemente ao programa chamador.
- O programador deve especificar os threads, associando-os às sub-rotinas assíncronas. Desta forma, um ambiente multithread possibilita a execução concorrente de sub-rotinas dentro de um mesmo processo.

Ambiente Multithread



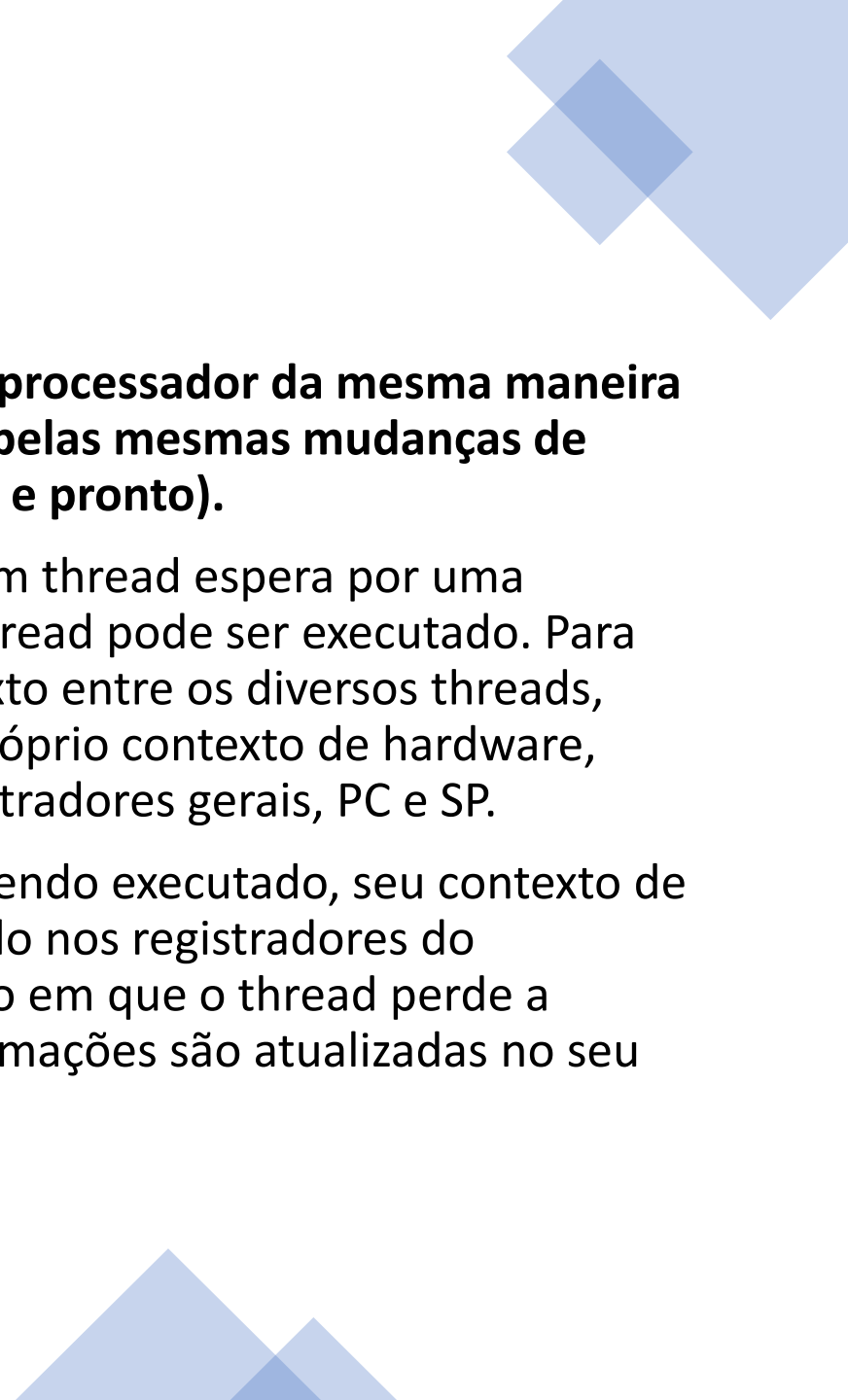


Ambiente Multithread

- **No ambiente multithread, cada processo pode responder a várias solicitações concorrentemente ou mesmo simultaneamente, caso haja mais de um processador.**
 - **A grande vantagem no uso de threads é a possibilidade de minimizar a alocação de recursos do sistema, além de diminuir o overhead (sobrecarga) na criação, troca e eliminação de processos.**
- 

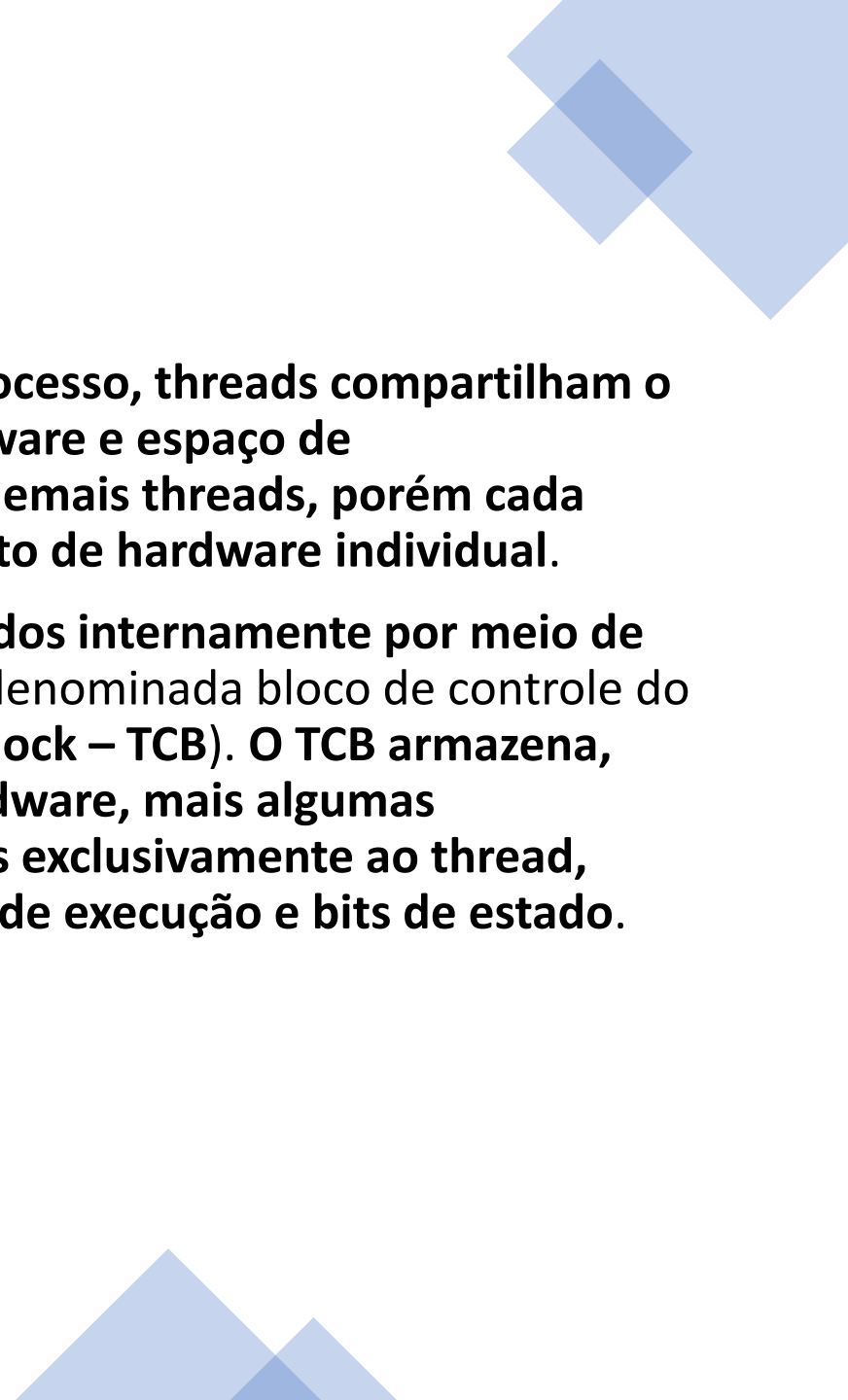


Ambiente Multithread

- **Threads compartilham o processador da mesma maneira que processos e passam pelas mesmas mudanças de estado (execução, espera e pronto).**
 - Por exemplo, enquanto um thread espera por uma operação de E/S, outro thread pode ser executado. Para permitir a troca de contexto entre os diversos threads, cada thread possui seu próprio contexto de hardware, com o conteúdo dos registradores gerais, PC e SP.
 - Quando um thread está sendo executado, seu contexto de hardware está armazenado nos registradores do processador. No momento em que o thread perde a utilização da UCP, as informações são atualizadas no seu contexto de hardware.
- 

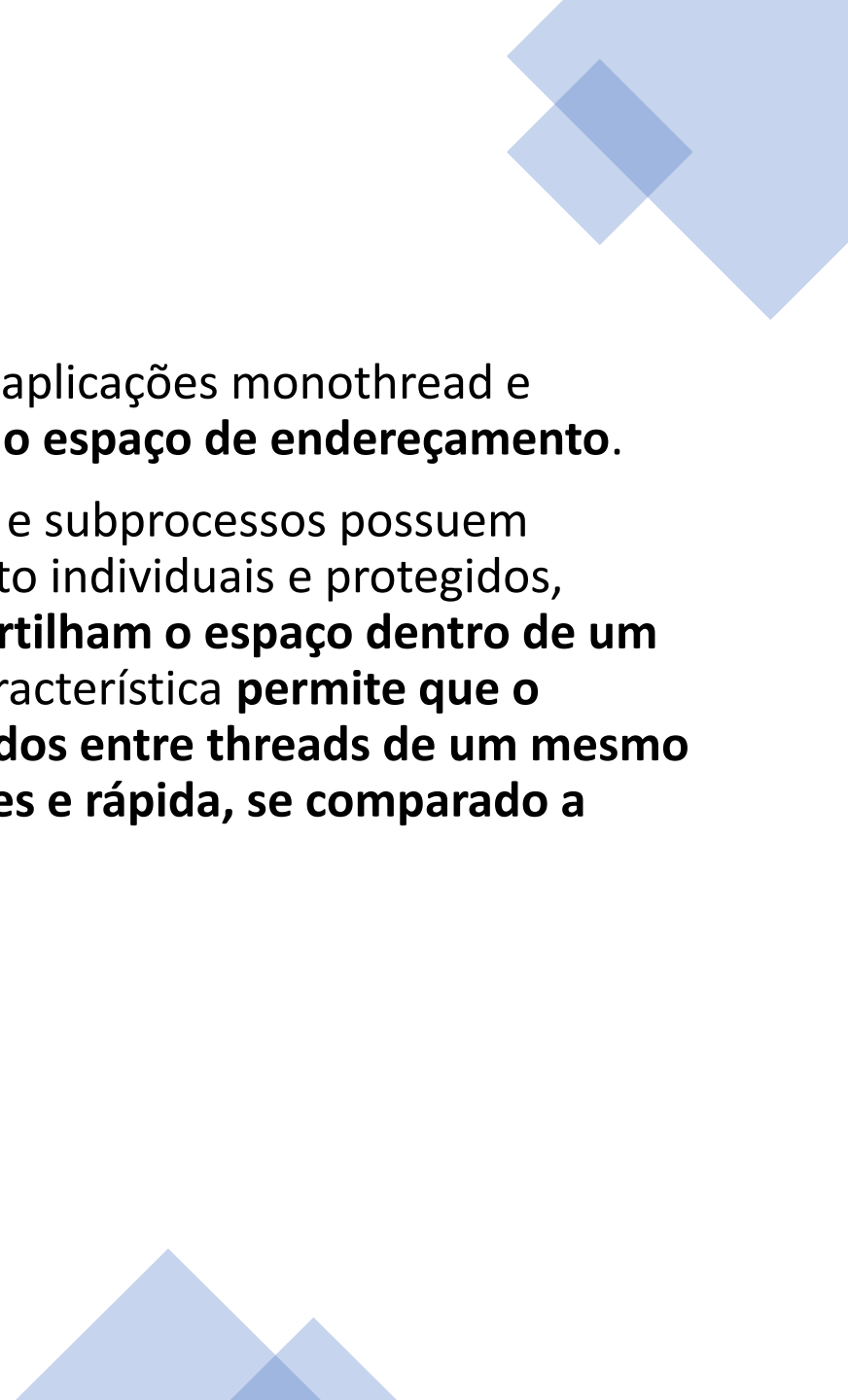


Ambiente Multithread

- **Dentro de um mesmo processo, threads compartilham o mesmo contexto de software e espaço de endereçamento com os demais threads, porém cada thread possui seu contexto de hardware individual.**
 - **Threads são implementados internamente por meio de uma estrutura de dados denominada bloco de controle do thread (Thread Control Block – TCB). O TCB armazena, além do contexto de hardware, mais algumas informações relacionadas exclusivamente ao thread, como prioridade, estado de execução e bits de estado.**
- 

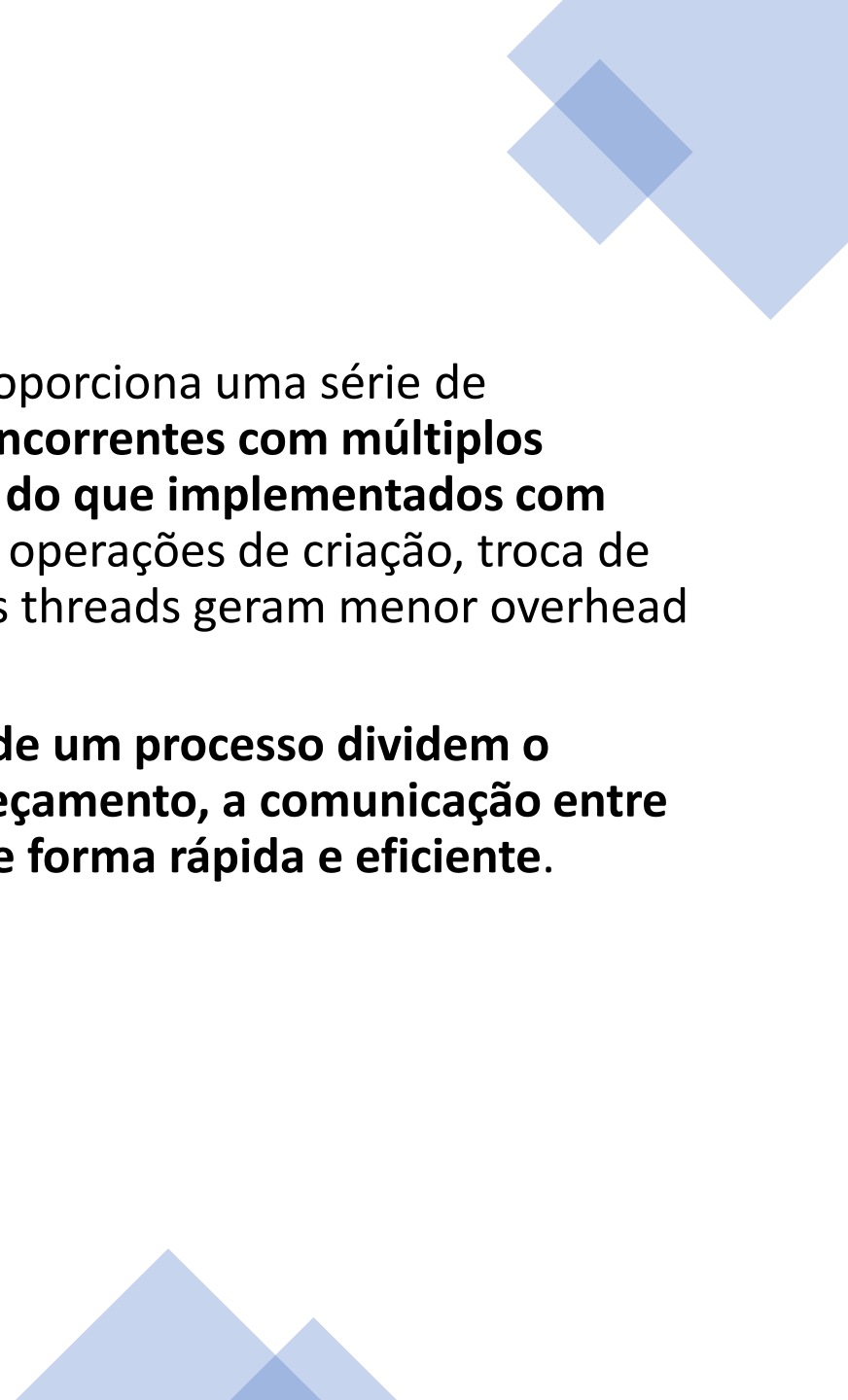


Monothread x Multithread

- A **grande diferença** entre aplicações monothread e multithread está no **uso do espaço de endereçamento**.
 - Processos independentes e subprocessos possuem espaços de endereçamento individuais e protegidos, enquanto **threads compartilham o espaço dentro de um mesmo processo**. Esta característica **permite que o compartilhamento de dados entre threads de um mesmo processo seja mais simples e rápida, se comparado a ambientes monothread**.
- 

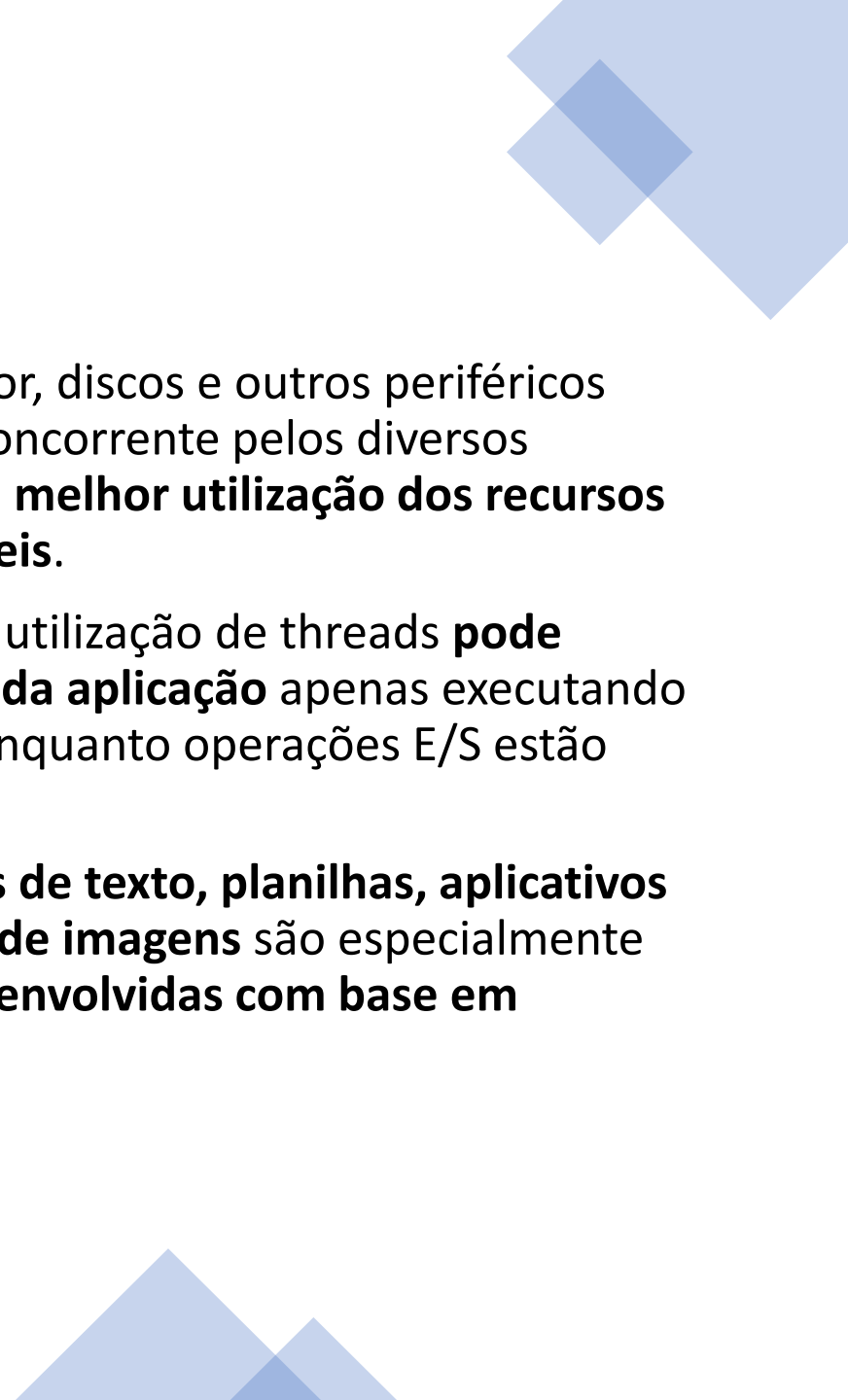


Monothread x Multithread

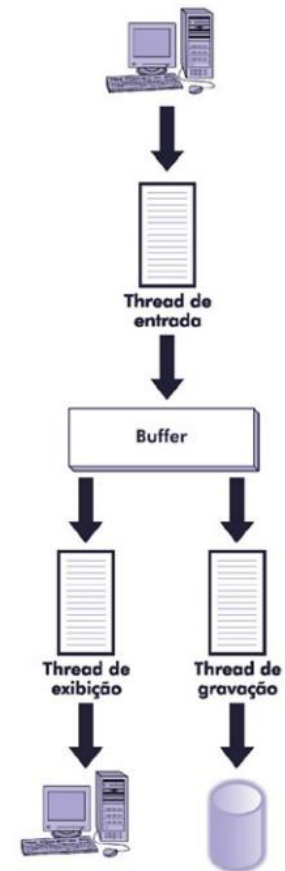
- O uso de **multithreads** proporciona uma série de **benefícios**. **Programas concorrentes com múltiplos threads são mais rápidos do que implementados com múltiplos processos**, pois operações de criação, troca de contexto e eliminação dos threads geram menor overhead (sobrecarga).
 - Como os **threads dentro de um processo dividem o mesmo espaço de endereçamento**, a comunicação entre eles pode ser realizada de forma rápida e eficiente.
- 



Monothread x Multithread

- A utilização do processador, discos e outros periféricos pode ser feita de forma concorrente pelos diversos threads, significando uma **melhor utilização dos recursos computacionais disponíveis**.
 - Em algumas aplicações, a utilização de threads **pode melhorar o desempenho da aplicação** apenas executando tarefas em background, enquanto operações E/S estão sendo processadas.
 - **Aplicações como editores de texto, planilhas, aplicativos gráficos e processadores de imagens** são especialmente beneficiadas quando desenvolvidas com base em threads.
- 

Aplicação multithread



Monothread x Multithread

- **Em ambientes cliente-servidor, threads são essenciais para solicitações de serviços remotos.**
- **Em um ambiente monothread, se uma aplicação solicita um serviço remoto ela pode ficar esperando indefinidamente, enquanto aguarda pelo resultado. Em um ambiente multithread, um thread pode solicitar o serviço remoto, enquanto a aplicação pode continuar realizando outras atividades.** Já para o processo que atende a solicitação, múltiplos threads permitem que diversos pedidos sejam atendidos simultaneamente.

Aplicação multithread

