

# TAREFAS E PROCESSOS EM SISTEMAS OPERACIONAIS

---

Lucas Silva de Oliveira

Bacharel em Sistemas de Informação

# CONTENTS

---

1. O Contexto Inicial: Programa vs. Tarefa
2. De Tarefa para Processo
3. Gerência de Tarefas
4. Troca de Contexto (Context Switch)
5. A Estrutura do Processo (O Contêiner)
6. Conclusão

# O CONTEXTO INICIAL: PROGRAMA VS. TAREFA

---

- **O que é um Programa?**

- Conceito estático.
- Arquivo no disco com instruções passivas (ex: `browser.exe`).
- **Receita de bolo** escrita no papel.

- **O que é uma Tarefa (Task)?**

- Conceito dinâmico.
- A execução sequencial de instruções com uma finalidade.
- Possui estado interno que muda ao longo do tempo.
- O ato de **cozinhar** seguindo a receita.

# DE TAREFA PARA PROCESSO

---

- **Como o SO gerencia Tarefas?**

- Ele precisa criar um ambiente completo para executá-las.

- **Conceito de Processo**

- É a forma como o SO implementa e isola uma tarefa.
- Processo = contêiner de recursos.
- Agrupa tudo o que a tarefa precisa: memória, arquivos, contexto de hardware.

- **Evolução**

- Antigamente: **1 processo = 1 tarefa**.
- Hoje: **1 processo pode conter várias tarefas** (threads).

### Pra entender melhor - 1

Segundo [1], o computador não executa arquivos, mas sim fluxos de instruções. Uma receita é o programa, o cozinheiro é o processamento (CPU) e os ingredientes são os dados de entrada. O processo é a atividade do cozinheiro ler a receita, buscar os ingredientes e preparo da receita pelo cozinheiro.

### Pra entender melhor - 2

O processo é a “caixa” que isola a tarefa para que ela não interfira em outras. Ele é o mecanismo do Sistema Operacional para encapsular execução.

# GERÊNCIA DE TAREFAS

---



- O processador tem de executar todas as tarefas submetidas pelos usuários. Essas tarefas geralmente têm comportamento, duração e importância distintas. Cabe ao sistema operacional organizar as tarefas para executá-las e decidir em que ordem fazê-lo [2].
- **Sistemas Monotarefas**
  - Anos 40.
  - Executa uma tarefa por vez.
  - Cada binário era carregado do disco para a memória e executado até o fim.
  - Os dados de entrada da tarefa eram carregados na memória junto à mesma e os resultados obtidos eram escritos no disco após a conclusão da tarefa.
  - Todas as operações de transferência de código e dados entre o disco e a memória eram coordenados por um operador humano.
  - Simples, mas ineficiente (CPU ociosa durante I/O).



Figure 1: Estados de uma tarefa em um sistema monotarefa.

*Adaptado de [2] (2019).*

- **Sistemas Multitarefas**

- Descompasso significativo entre a alta velocidade de processamento e a baixa velocidade dos dispositivos de entrada/saída, o que deixava o processador ocioso durante transferências de dados.
- O monitor de sistema operacional foi introduzido para gerenciar múltiplas tarefas.
- Suspender tarefas que aguardam I/O para permitir que outras sejam executadas.
- Passar o processador para outra tarefa pronta para execução.
- Permitia que o processador fosse compartilhado entre várias tarefas, alternando entre elas conforme necessário.

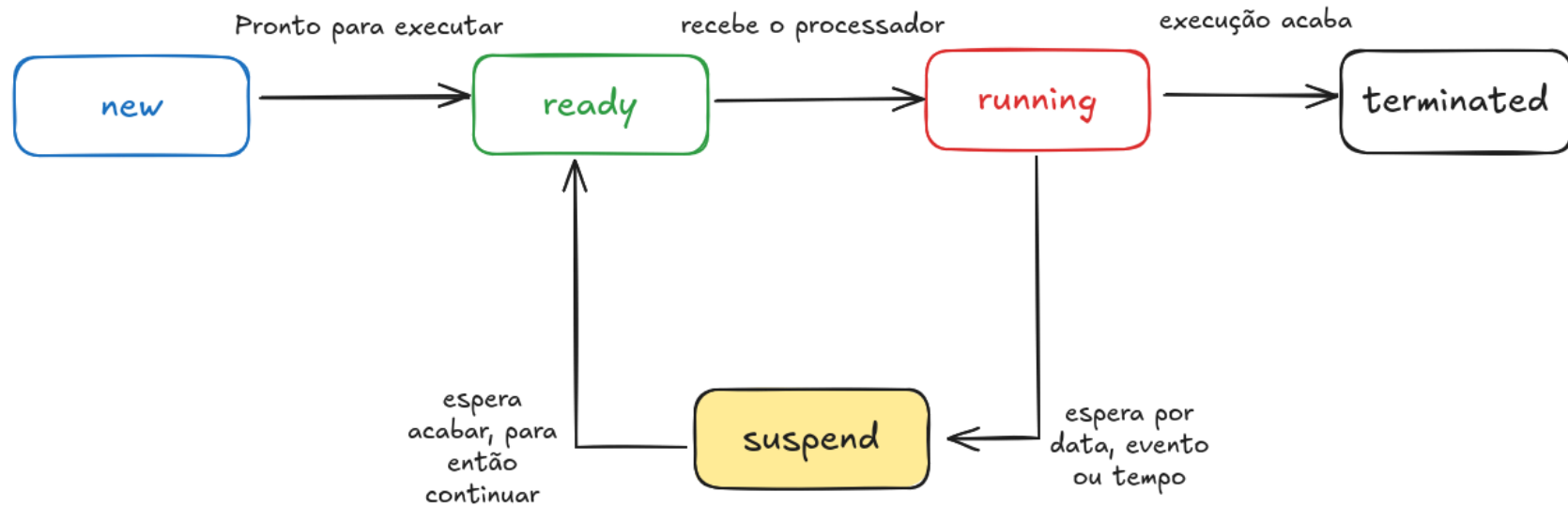


Figure 2: Diagrama de estados de uma tarefa em um sistema multitarefas.

*Adaptado de [2] (2019).*

- **Sistemas de Tempo Compartilhado**

- O processador é dividido em pequenos intervalos de tempo (quantum).
- Cada tarefa recebe um quantum para executar antes de ser interrompida (Preempção) e passar o processador para outra tarefa.

- ▶ A implementação depende de um temporizador programável no hardware, que gera interrupções periódicas (chamadas de ticks). Essas interrupções devolvem o controle ao núcleo do sistema operacional regularmente.
- ▶ Quando uma tarefa assume o processador, o núcleo define um contador de ticks (baseado no quantum). A cada interrupção do relógio, o contador é decrementado. Se chegar a zero, a tarefa sofre preempção.
- ▶ Cria a ilusão de multitarefa real, onde várias tarefas parecem ser executadas ao mesmo tempo.

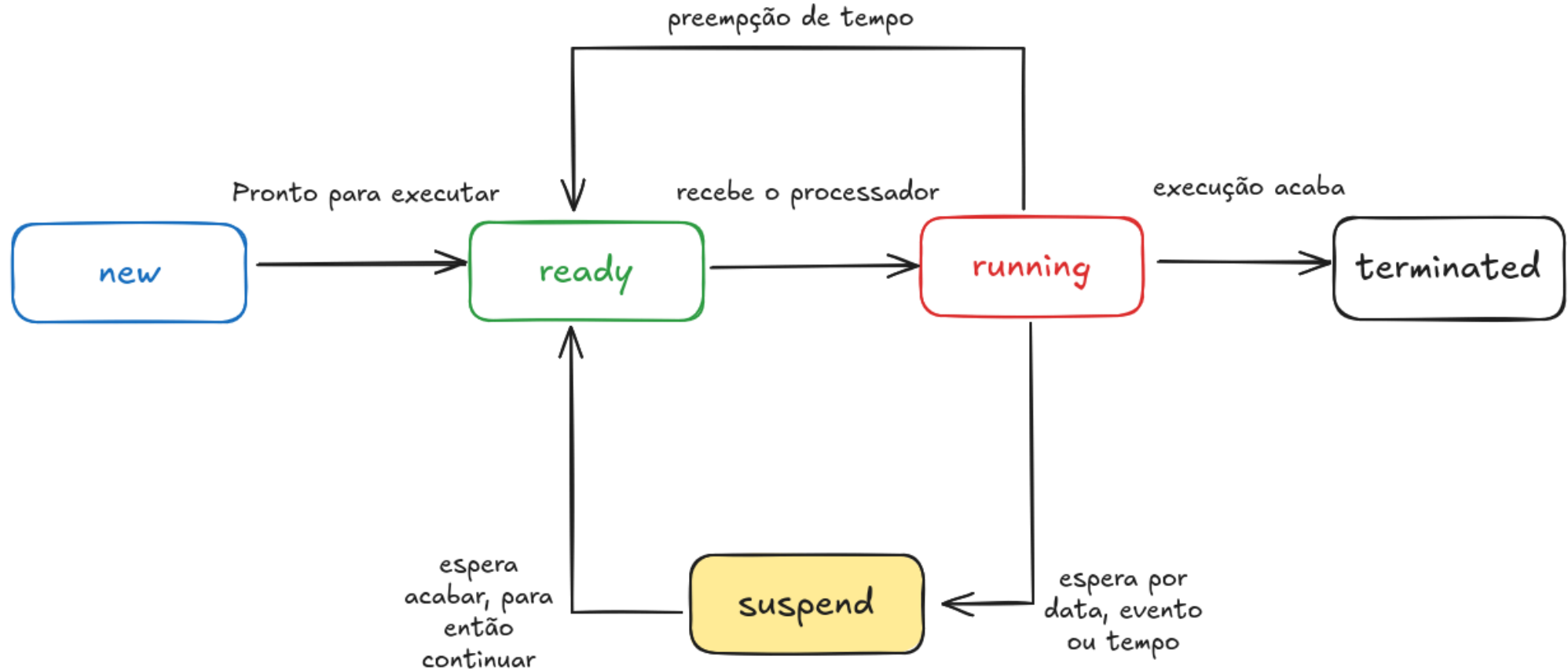


Figure 3: Diagrama de estados de uma tarefa em um sistema de tempo compartilhado.  
*Adaptado de [2] (2019).*

# TROCA DE CONTEXTO (CONTEXT SWITCH)

---

- **Definição:**

- Para suspender uma tarefa e retomar outra, o SO deve salvar e restaurar seu contexto. O contexto diz respeito a estado atual de uma tarefa: registradores, variáveis, contador de programa, stack pointer etc.

- **Mecanismo:**

1. Salvar contexto, o associando a um descritor, uma estrutura de dados no núcleo do SO que se chama TCB(Task Control Block).
2. Escalonador escolhe próxima tarefa.
3. Restaurar contexto no processador.

- **Custo:**

- Consome tempo do processador → overhead.

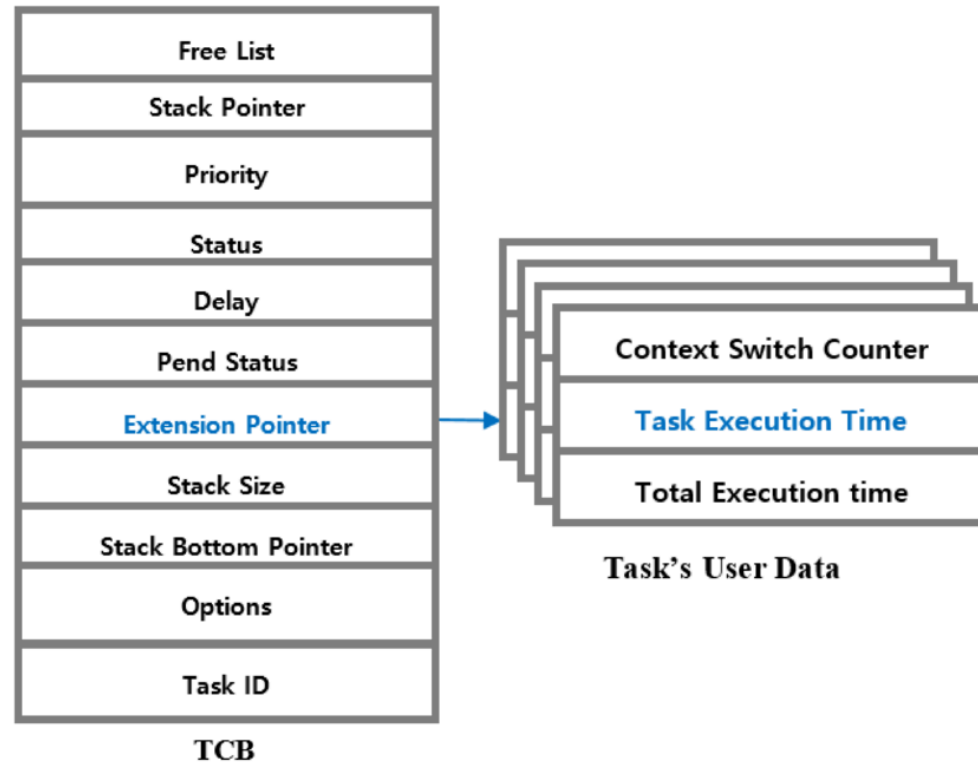


Figure 4: Structure of the Task Control Block.

Fonte: [3]. (2025).



# A ESTRUTURA DO PROCESSO (O CONTÊINER)

---

- O processo é o contêiner que agrupa todos os recursos necessários para a execução de uma ou mais tarefas.
- **O que há dentro de um Processo?**
  1. Espaço de Endereçamento → código, dados, pilha.
  2. Contexto de Hardware → registradores (PC, SP...).
  3. Recursos do Núcleo → arquivos abertos, conexões etc.
- Os processos são isolados entre si pelos mecanismos de proteção providos pelo hardware (isolamento de áreas de memória, níveis de operação e chamadas de sistema), impedindo que uma tarefa do processo  $p|a$  acesse um recurso atribuído ao processo  $p|b$ .
- Um processo pode conter múltiplas tarefas (threads) que compartilham o mesmo espaço de endereçamento e recursos, mas possuem seus próprios contextos de hardware.

### Pra entender melhor - 3

O núcleo do sistema operacional mantém descritores de processos, denominados PCBs (Process Control Blocks), para armazenar as informações referentes aos processos ativos. Um PCB contém informações como o identificador do processo (PID – Process IDentifier), seu usuário, prioridade, data de início, caminho do arquivo contendo o código executado pelo processo, áreas de memória em uso, arquivos abertos, etc

# CONCLUSÃO

---

- **Resumo:**
  - Tarefa → fluxo dinâmico de execução.
  - Processo → contêiner que provê recursos e isolamento.
  - Estados → Pronta, Executando, Suspensa.
  - Context Switch → permite multitarefa real.
- **Importância:**
  - Gerência eficiente mantém o sistema responsivo e estável.

# REFERENCES

---

- [1] A. S. Tanenbaum, “Sistemas Operacionais Modernos.” Pearson Education, São Paulo, 2015.
- [2] C. A. Maziero, “Sistemas Operacionais: Conceitos e Mecanismos.” Editora da UFPR, 2019.
- [3] S. Han, K. Lee, S.-J. Cho, and M. Park, “Anomaly Detection Based on Temporal Behavior Monitoring in Programmable Logic Controllers,” *Electronics*, vol. 10, p. 1218, 2021, doi: [10.3390/electronics10101218](https://doi.org/10.3390/electronics10101218).