# 

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA**

**PARAÍBA – CAMPUS *ESPERANÇA***

**SISTEMAS OPERACIONAIS**

**ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**Relatório Técnico:**

**Leitores/Escritores com Buffer Circular e Sincronização via Semáforo.**

Thiago Silva Bezerra  
Renato Medeiros Pereira

João Francisco da Silva Monte

Miguel Arcanjo Targino Valentim

## 1. Introdução

Este relatório descreve a implementação e os conceitos envolvidos na simulação do problema clássico de **Leitores/Escritores**, utilizando **buffer circular**, **semáforos** e **mutex** para garantir a **sincronização** adequada entre processos concorrentes.

## 

## 2. Objetivo

Demonstrar a gestão de processos concorrentes através da implementação de uma solução onde múltiplos **produtores (escritores)** e **consumidores (leitores)** compartilham o acesso a um recurso comum — um buffer circular — de forma segura e eficiente.

## 3. Conceitos Fundamentais

### 3.1. Produtores e Consumidores

* **Produtores (Escritores):** Criam itens e os inserem no buffer se houver espaço.
* **Consumidores (Leitores):** Retiram itens do buffer em ordem FIFO, caso haja itens disponíveis.

### 3.2. Semáforos

Utilizados para controlar o acesso ao buffer e garantir exclusão mútua:

* **Mutex:** Garante que apenas um processo acesse a região crítica por vez.
* **Semáforo “vazios” (emptySlots):** Conta os espaços disponíveis no buffer.
* **Semáforo “cheios” (fullSlots):** Conta os itens disponíveis para consumo.

### 3.3. Buffer Circular

Implementado logicamente através de um array com tamanho fixo, que é reutilizado continuamente, garantindo eficiência no uso de memória e desempenho na comunicação entre threads.

## 

## Arquitetura da Solução

### 4.1. Diagrama de Componentes

|  |
| --- |
| index.js |
| Simula produtores e consumidores |

|  |
| --- |
| AcessoConcorrenteAoBuffer |
| buffer[]  mutex  emptySlots  fullSlots |

|  |
| --- |
| MUTEX |

|  |
| --- |
| SEMAPHORE |

### │

### 

### 4.2. Ciclo de Produção e Consumo

|  |  |
| --- | --- |
| **PRODUTOR** | **CONSUMIDOR** |
| Produz item | Aguarda item |
| emptySlots.adquire() |  |
| mutex.adquire() | fullSlots.adquire() |
| Insere no buffer | mutex.adquire() |
| mutex.release() | Remove do buffer |
| fullSlots.release() | mutex.release() |
|  | emptySlots.release() |

## 5. Ciclo de Execução

### Produção:

1. Verifica se há espaço disponível (emptySlots).
2. Adquire o mutex para entrar na região crítica.
3. Insere item no buffer.
4. Libera o mutex.
5. Sinaliza item disponível (fullSlots).

### Consumo:

1. Verifica se há itens disponíveis (fullSlots).
2. Adquire o mutex para entrar na região crítica.
3. Remove item do buffer.
4. Libera o mutex.
5. Sinaliza espaço liberado (emptySlots).

## 6. Benefícios e Resultados

* **Segurança:** O uso de mutex e semáforos impede condições de corrida.
* **Eficiência:** O buffer circular permite alta taxa de transferência de dados.
* **Escalabilidade:** A arquitetura suporta múltiplos leitores e escritores sem interferência.

## 7. Aplicacoes Práticas

* **Sistemas Operacionais:** Gerenciamento de threads e I/O.
* **Automacão Industrial:** Coordenação de sensores e atuadores.
* **Bancos de Dados:** Controle de concorrência de transações.
* **Computação Distribuída:** Comunicação entre nós.

## 8. Conclusão

A implementação da solução de Leitores/Escritores utilizando semáforos e buffer circular se mostrou eficaz na demonstração de sincronização e exclusão mútua. Com essa abordagem, garantimos consistência de dados e comunicação segura entre processos concorrentes, refletindo conceitos fundamentais de sistemas operacionais.