

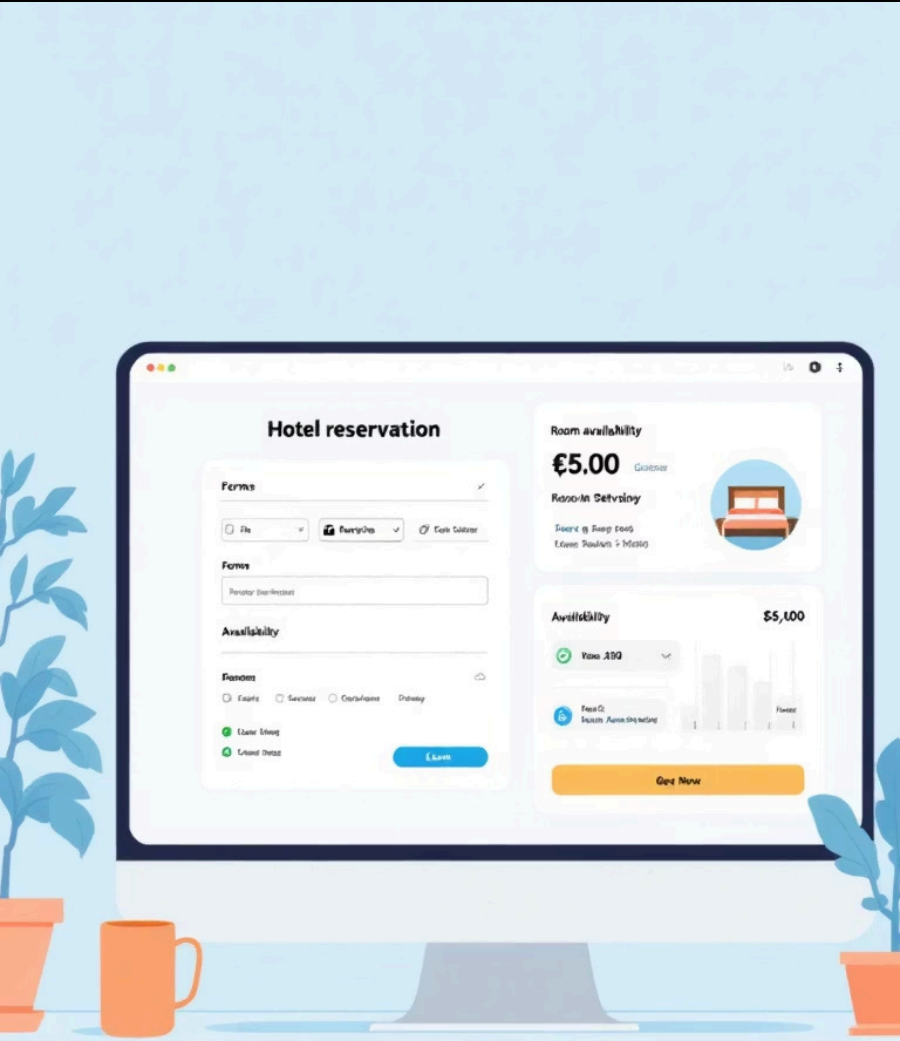
# Avaliação de Desempenho: Sistema de Reservas de Hotel

## Comparação de Mecanismos de Sincronização em Go

**Autores:** Mikael Cavalcanti e Lucas Monterazo

**Data:** 1 de Outubro 2025

**Instituição:** UFPE - Programação Concorrente e Distribuída



# PASSO - OBJETIVO DA AVALIAÇÃO

**Objetivo Principal:** Comparar o desempenho de um sistema de reservas de hotel utilizando dois mecanismos de sincronização em Go

Soluções Comparadas:

## Solução A

Atomic Operations + RWMutex

## Solução B

Canais (Monitor Pattern)

## Questão de Pesquisa:

"Qual mecanismo oferece melhor desempenho para gerenciar recursos compartilhados em um sistema de reservas concorrente?"

# PASSO - SERVIÇOS DO SISTEMA

## Sistema Avaliado: Aplicação de Reservas de Hotel

### Serviços Oferecidos:

- Reserva de quartos (4 tipos disponíveis)
- Cancelamento de reservas
- Consulta de disponibilidade

### Tipos de Quartos:

15

Standard

8

Luxo

5

Suite

2

Presidencial

### Resultados Possíveis:

✓ Reserva confirmada × Reserva negada (indisponibilidade) ↺ Cancelamento processado

# PASSO – MÉTRICAS DE DESEMPENHO

Métrica Principal:

## Tempo médio total de execução (milissegundos)

Métricas Secundárias:



❏ **Justificativa:** Métricas focadas em velocidade e eficiência de execução das threads.



# PASSO - PARÂMETROS DO SISTEMA

## Parâmetros do Sistema:

### Hardware:

- Processador: Intel(R) Core(TM) i7-14700HX (2.10 GHz)
- Memória: 16GB RAM
- SO: Windows 11

### Software:

- Linguagem: Go 1.21+

## Parâmetros da Carga de Trabalho:

- Primeira onda: 40 clientes concorrentes
- Cancelamentos: 5 operações simultâneas
- Segunda onda: 10 clientes concorrentes
- Tempo de processamento: 0-150ms (aleatório)
- Total de quartos: 30 unidades

# PASSO - FATORES E NÍVEIS

## Fator Avaliado: Mecanismo de Sincronização

Níveis:

1

### Atomic Operations + RWMutex

- Operações atômicas para contadores
- RWMutex para proteção do mapa de quartos
- Acesso concorrente otimizado

2

### Canais (Monitor Pattern)




- Goroutine centralizadora (Gerente de Reservas)
- Comunicação via canais
- Serialização de acesso ao estado

📌 **Parâmetro Fixo:** GOMAXPROCS = 6 (todas as execuções)

# PASSO - TÉCNICA DE AVALIAÇÃO

## Técnica Escolhida: Medição

### Justificativa:

-  Sistema real implementado e funcional
-  Permite medição direta e precisa
-  Resultados confiáveis e reproduzíveis

### Ferramentas Utilizadas:

- `time.Now()` e `time.Since()` (Go)
- 5 execução por configuração

### Vantagens da Medição:

- ✓ Resultados reais do sistema

# PASSO - CARGA DE TRABALHO

## Carga Sintética Representativa:

### Fase 1 - Primeira Onda:

- 40 clientes simultâneos
- Requisições de reserva

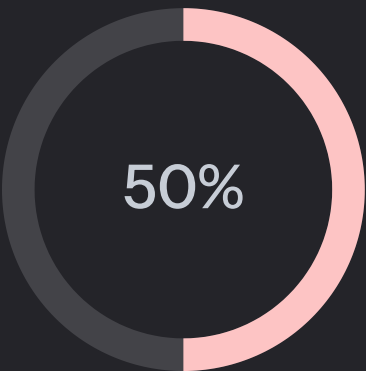
### Fase 2 - Cancelamentos:

- 5 cancelamentos concorrentes
- Liberação de recursos

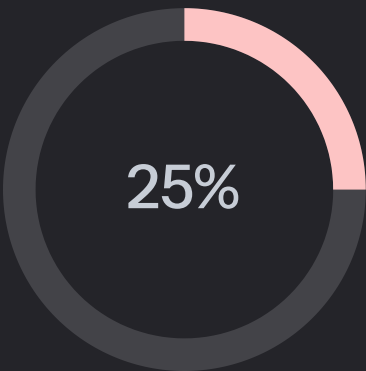
### Fase 3 - Segunda Onda:

- 10 clientes simultâneos
- Aproveitamento de vagas liberadas

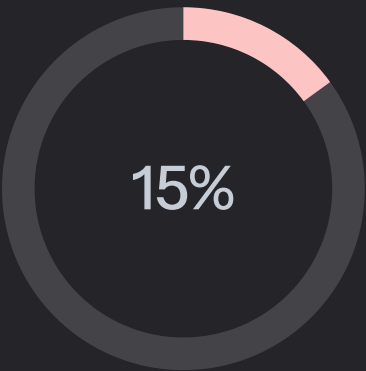
## Distribuição de Preferências:



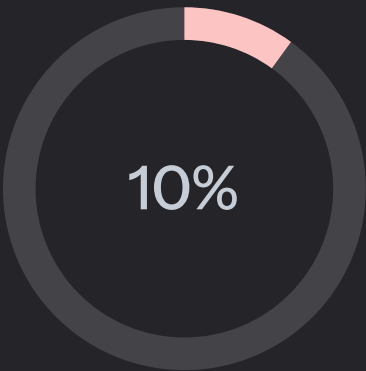
Quartos Standard



Quartos Luxo



Quartos Suite



Quartos Presidencial

## Características:

- Seed aleatório por execução
- Simula cenário realista de hotel
- Competição por recursos limitados

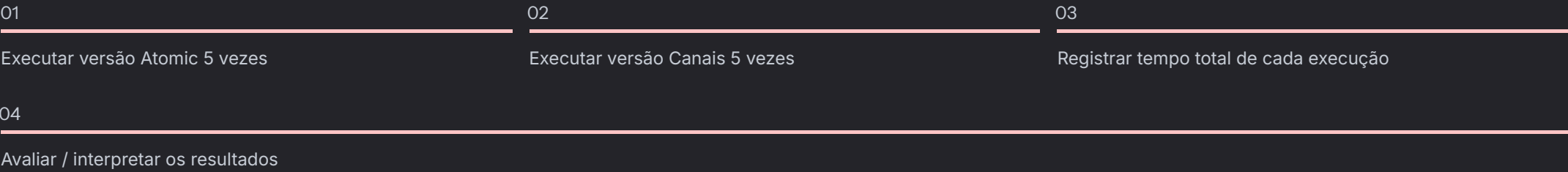


# PASSO - GRÁFICO DO EXEPRIMENTO

## Metodologia:

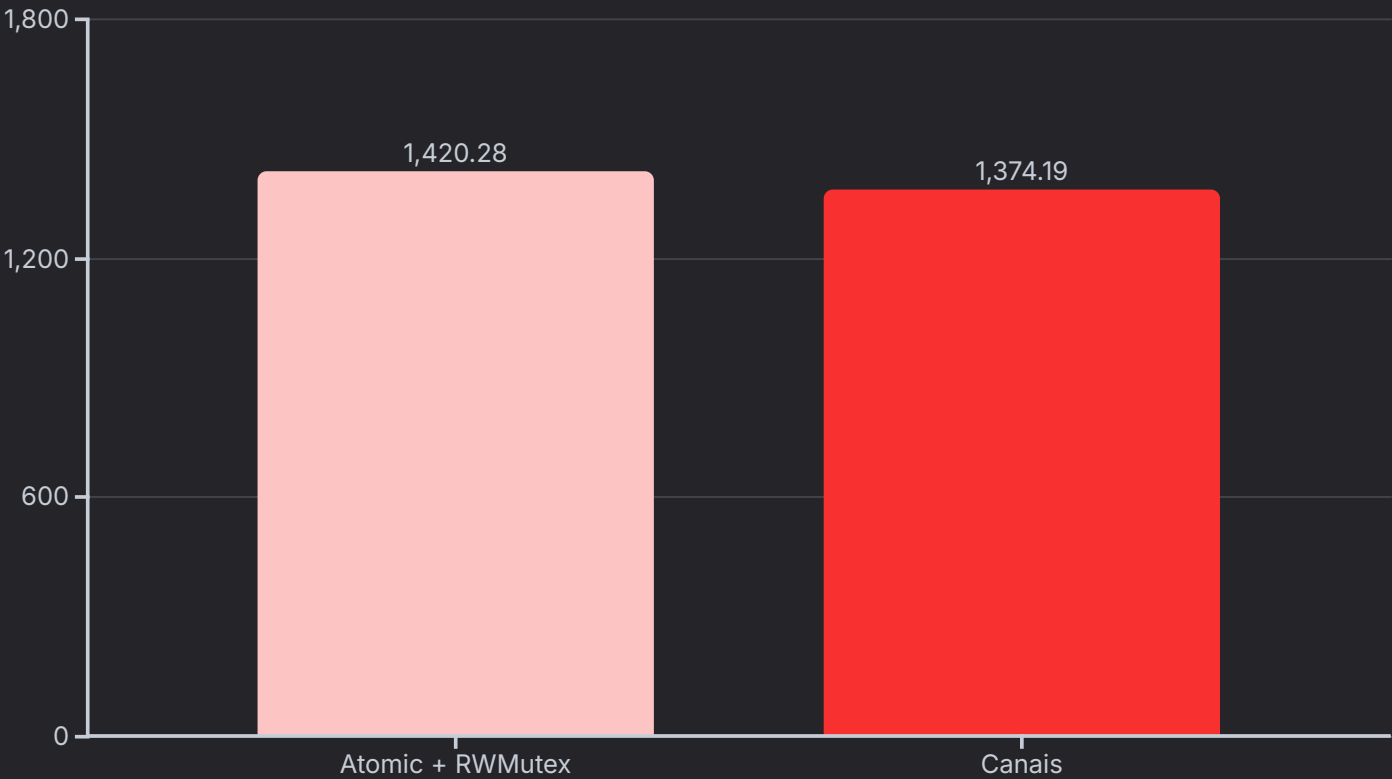
- 5 execuções por configuração
- Total: 10 execuções

## Procedimento:



## Controle de Variáveis:

✓ Mesma máquina ✓ Mesmo ambiente ✓ Mesma carga de trabalho ✓ Ordem de execução aleatória



Título: "Tempo Médio de Execução por Mecanismo"

# PASSO - INTERPRETAÇÃO DOS RESULTADOS

Observamos que o Mutex apresentou uma performance melhor do que o chanel, tendo em vista que o Mutex é um tipo de implementação mais simples em comparação (basicamente a manipulação de um inteiro) enquanto o chanel que possui uma implementação mais abstrata além de conter o Mutex implicitamente em sua implementação.