

# Relatório de Laboratório: Qualidade de Serviço (QoS) - A Otimização da Jornada dos Pacotes

---

**Disciplina:** Redes de Computadores II **Professora:** Angelita Rettore de Araujo

**Nome do Aluno:** Felipe Biava Favarin

**Turma:** Redes II - 6º Fase

---

## 1. Introdução

Este laboratório aborda a **Qualidade de Serviço (QoS)**, um conjunto de mecanismos importantes para gerenciar o tráfego de rede e assegurar que aplicações críticas recebam tratamento preferencial. Diferente dos laboratórios anteriores que focaram na confiabilidade (garantir que os pacotes cheguem), o objetivo aqui é garantir que os pacotes cheguem *com qualidade* – ou seja, com a latência, jitter, throughput e perda de pacotes adequados.

A importância da QoS é contextualizada pela **narrativa da telecirurgia**, onde cada pacote de comando tátil, voz ou dado vital do paciente é crucial. Atrasos, variações irregulares na chegada ou perda de pacotes podem ter consequências catastróficas.

## 2. Objetivos

Os principais objetivos deste laboratório são:

1. **Compreender e medir** os conceitos fundamentais de Latência, Jitter, Throughput, Perda de Pacotes e Classificação de Tráfego no contexto de QoS.
2. **Configurar e executar simulações** no **Network Simulator 2 (NS2)** para observar o comportamento da rede sob diferentes condições de QoS.
3. **Utilizar o Wireshark** para capturar e analisar o tráfego de rede, medindo parâmetros de QoS em tempo real.
4. **Analisar o impacto** da variação dos parâmetros de QoS no desempenho de diferentes tipos de aplicações.
5. **Comparar a tolerância a perdas e a sensibilidade à latência e jitter** de diversas aplicações.
6. **Propor soluções** baseadas em QoS para otimizar o desempenho de aplicações críticas em cenários de rede desafiadores.

## 3. Ferramentas Utilizadas

- **Network Simulator 2 (NS2):** Ambiente de simulação de rede para modelar cenários.
  - **Wireshark:** Analisador de protocolo de rede para captura e inspeção de pacotes em tempo real.
  - **Acesso à Internet:** Para testes com ferramentas online (como Google Meet).
- 


## 4. Parte I: Relembrando a Jornada – Preparando o Ambiente

**Contexto Teórico:** A narrativa da cirurgia remota é a base para entender a importância dos "pacotes heróis" (Pablo, Melody, Flash e Data) e como a QoS é vital para a missão deles de salvar uma vida.

#### 4.1. Verificação e Configuração Inicial do NS2

- Confirmei a instalação do NS2 e criei o arquivo `qos_base.tcl`.


**Entrega:** Captura de tela do `qos_base.tcl` no editor de texto.

 captura de tela do arquivo qos\_base

#### 4.2. Configuração Inicial do Wireshark

- Abri o Wireshark e selecionei a interface de rede correta para captura.

**Entrega:** Captura de tela do Wireshark com a interface de captura selecionada.

 captura de tela do Wireshark mostrando a interface selecionada

---

### 5. Parte II: Latência (Delay) – O Tempo é Essencial

**Contexto Teórico:** A latência é o tempo que um pacote leva para ir da origem ao destino, como o tempo para o comando tátil do Dr. Martinez (Flash) chegar ao bisturi em Manaus.

#### 5.1. Simulação de Latência no NS2

- Criei e executei o script `lab_latencia.tcl`, experimentando diferentes valores para `link_delay` (ex: 10ms, 100ms, 500ms).

**Entrega:** O código `lab_latencia.tcl` utilizado.

```
# [Conferir arquivo lab_latencia.tcl]
```

#### 5.2. Análise da Latência no Arquivo de Trace (.tr)

- Analisei o arquivo `lab_latencia.tr`, identificando o envio e recebimento de pacotes para calcular a latência de ponta a ponta.

**Entrega:** Trecho do arquivo `.tr` destacando um pacote enviado e seu respectivo recebimento.

### -#- Trecho do Arquivo -#- --#

---

O arquivo gerado foi o `base_output.tr`

Nele temos:

- 0.51 0 1 cbr 1000 ----- 0 0.0 1.0 1 1
- 0.51 0 1 cbr 1000 ----- 0 0.0 1.0 1 1 r 1.018 0 1 cbr 1000 ----- 0 0.0 1.0 1 1

Onde: O último número indica o id do pacote, nesse caso, sobre o pacote 1 "+" significa que o pacote está enfileirado, "-" significa que o pacote foi retirado da fila, e "r" significa a receber (na saída do link), isso é, a comprovação que o pacote foi recebido.

### Cálculos da Latência:

Cálculo da latência:  $\text{latência} = (\text{tempo de recebimento} - \text{tempo de envio})$

## Pacotes Selecionados

---

10ms:

- 0.69 0 1 cbr 1000 ----- 0 0.0 1.0 19 19
- 0.69 0 1 cbr 1000 ----- 0 0.0 1.0 19 19 r 0.708 0 1 cbr 1000 ----- 0 0.0 1.0 19 19

100ms:

- 0.85 0 1 cbr 1000 ----- 0 0.0 1.0 35 35
- 0.85 0 1 cbr 1000 ----- 0 0.0 1.0 35 35 r 0.958 0 1 cbr 1000 ----- 0 0.0 1.0 35 35

500ms:

- 0.77 0 1 cbr 1000 ----- 0 0.0 1.0 27 27
- 0.77 0 1 cbr 1000 ----- 0 0.0 1.0 27 27 r 1.278 0 1 cbr 1000 ----- 0 0.0 1.0 27 27

| link_delay Configurado  | Timestamp Envio | Timestamp Recebimento | Latência Calculada |
|-------------------------|-----------------|-----------------------|--------------------|
| [Valor 1 (e.g., 10ms)]  | [0,69]          | [0,708]               | [0,018]            |
| [Valor 2 (e.g., 100ms)] | [0,85]          | [0,958]               | [0,108]            |
| [Valor 3 (e.g., 500ms)] | [0,77]          | [1,278]               | [0,508]            |

### 5.3. Perguntas para Refletir e Discutir

#### 1. Qual a relação entre o link\_delay configurado no script e a latência medida no arquivo .tr?

- Ao mudarmos o link\_delay para 10ms vemos que os pacotes de tamanho 1000 bytes vão mais rapidamente pelo canal e o receptor começa a receber antes, o delay de envio do 1º pacote ao recebimento do mesmo é menor, com 10ms temos uma latência de 0,018 segundos. Mudando para 100ms de link\_delay, temos vários pacotes de 1000 bytes jogados no canal e o tempo de latência aumenta para 0,108 segundos. E quanto mais aumentamos o link\_delay mais vemos um impacto direto na latência, com 500ms temos 0,508 de latência.

#### 2. Como a latência afeta a percepção do usuário em aplicações como VoIP ou jogos online?

- A demora para o recebimento de pacotes pode fazer com que o jogador1 (num jogo online) comece a travar constantemente, pois é o tempo de recebimento + processamento do computador dele. Considerando um jogador2 com latência menor, ele não passará pelas

mesmas dificuldades que o outro jogador<sup>1</sup>. Numa chamada de vídeo (VoIP), temos a perda / travamento da voz e interferências.

### 3. Se o Dr. Martinez estivesse em Tóquio e o paciente em Manaus, qual seria o impacto na latência?

- Considerando a distância em que os pacotes da comunicação terão que passar por vários roteadores, servidores e assim por diante. Além do tempo de processamento de cada, junto com o delay dos links (conexões) teremos um atraso enorme para a telecirurgia. Praticamente, tornando a cirurgia inviável.

---

## 6. Parte III: Jitter e Perda de Pacotes – A Variação Inesperada e o Preço da Imperfeição


**Contexto Teórico: Jitter** é a variação no atraso dos pacotes, causando "voz robotizada" (pacotes de Melody). A **perda de pacotes** ocorre quando um pacote não chega, sendo a tolerância variável por aplicação (pacotes de Data). O **RTCP (Real-Time Control Protocol)** é utilizado por aplicações em tempo real (como Google Meet) para reportar a qualidade da transmissão, incluindo jitter e perda.

### 6.1. Análise do Jitter e Perda de Pacotes no Wireshark (Captura Local de RTCP)


- Iniciei uma chamada no Google Meet e capturei o tráfego com o Wireshark.
- Filtrei o tráfego por **rtcp** e identifiquei os tipos de pacotes (SR, RR, SDES, Bye).
- Analisei os **Receiver Reports (RR)** para localizar os campos **Fraction Lost**, **Cumulative Number of Packets Lost** e **Interarrival Jitter**.

#### Entregas:

1. Captura de tela do Wireshark mostrando a captura inicial de pacotes.


 captura de tela da captura inicial de pacotes no Wireshark

2. Captura de tela do Wireshark mostrando o filtro **rtcp** aplicado.

 captura de tela da captura inicial do Wireshark com o filtro selecionado

3. Captura de tela dos detalhes de um pacote **Receiver Report (RR)**, com os campos **Fraction Lost**, **Cumulative Number of Packets Lost** e **Interarrival Jitter** claramente visíveis.

 captura de tela da captura inicial do Wireshark com os campos jitter, fraction lost e cumulative visíveis

 captura de tela da captura inicial do Wireshark para confirmação dos campos jitter, fraction lost e cumulative

#### Valores Observados:

- **Interarrival Jitter:** 1550922407 ms -> são quase 18 dias
- **Fraction Lost:**  $83 / 256 = 0,32422 \times 100 = 32,422\%$  (ou % se convertido)
- **Cumulative Number of Packets Lost:** 6701384

### 6.2. Perguntas para Refletir e Discutir

### 1. Como esses valores de Jitter e Fraction Lost se comparam aos limites aceitáveis para uma boa qualidade de voz/vídeo (ex: jitter idealmente abaixo de 30ms, perda abaixo de 1%)?

- Parece que a captura do valor de jitter realizado pelo WireShark apresenta problemas, pois se considerarmos o valor 1550922407 ms, são quase 17,95 dias. O que torna um pouco irreal essa análise. Considerando o valor de fraction lost também não temos o ideal, ele está em 32,422%.

Fiz o teste analisando outro pacote e esses foram os valores:

- **Interarrival Jitter:** 114427906 ms -> 1,3244 dias
- **Fraction Lost:**  $148 / 256 = 0,578125 \times 100 = 57,81\%$  (ou % se convertido)
- **Cumulative Number of Packets Lost:** -5804984

Realmente, não sei como o WireShark faz essa análise, mas suponho que esteja incorreta.

### 2. Por que o RTCP é essencial para aplicações em tempo real, mesmo que o RTP (dados de mídia) esteja criptografado?

- Porque no RTCP temos as informações de jitter e perdas, servindo como relatório. A partir desse relatório é possível ajustar as perdas e atrasos, diminuindo-os ao aceitável. Logo, ao operar sobre o protocolo TCP, pode-se alternar entre RTP e RTCP para a garantia de entrega dos pacotes.

### 3. Como as informações de jitter e perda de pacotes reportadas pelo RTCP podem ser usadas pela aplicação (Google Meet) para ajustar a qualidade da transmissão?

- O Google Meet avalia o tempo de chegada dos pacotes, percebe que estão chegando com um tempo cada vez maior, logo pode mandar diminuir o bitrate (mandar menos dados por segundo), a resolução do vídeo ou o frame rate (os fps - quadros por segundo), mesmo que signifique diminuir a qualidade da transmissão de vídeo para manter a qualidade de áudio.

---

## 7. Parte IV: Throughput vs. Responsividade – O Dilema da Rede

**Contexto Teórico: Throughput** é a quantidade de dados em um tempo (Pablo/vídeo HD), enquanto **responsividade** é a rapidez da resposta (Flash/comando tátil). Nem sempre é possível ter ambos em níveis máximos simultaneamente.

### 7.1. Simulação de Throughput e Responsividade no NS2

- Criei e executei o script `lab_throughput_responsividade.tcl`, comparando o comportamento de FTP (alto throughput) com Ping (alta responsividade).

**Entrega:** O código `lab_throughput_responsividade.tcl` utilizado.

```
# [Conferir arquivo lab_throughput_responsividade.tcl]
```

### 7.2. Análise do Throughput e Responsividade

- Analisei o arquivo `lab_throughput_responsividade.tr` para calcular o throughput do FTP e a latência de cada ping.

**Cálculos Detalhados do Throughput do FTP:**

- Número de pacotes TCP recebidos: 3702
- Tamanho do pacote TCP (padrão NS2): 512 bytes (ou especifique se diferente) 1040
- Tempo total da simulação para FTP (stop - start): [0.5 até 4.5 = 4] foram 4 segundos
- Throughput = (Número de pacotes \* Tamanho do pacote) / Tempo
- Throughput = (3702 \* 1040) / 4
- Throughput (em Kbps/Mbps): 962520 = 962,52Kbps

**Cálculos da Latência para cada pacote Ping e Impacto do FTP:**

Como calcular:

Latência = Timestamp Recebimento - Timestamp Envio

| Ping N° | Timestamp Envio | Timestamp Recebimento | Latência (ms) | Observações sobre o Impacto do FTP |
|---------|-----------------|-----------------------|---------------|------------------------------------|
| 1       | [1.0008]        | [1.010851]            | [0,010051]    | [Nenhum impacto significativo]     |
| 2       | [1.010851]      | [1.020902]            | [0,010051]    | [Nenhum impacto significativo]     |
| 3       | [1.020902]      | [1.030954]            | [0,010052]    | [Nenhum impacto significativo]     |
| 4       | [1.030954]      | [1.041005]            | [0,010051]    | [Nenhum impacto significativo]     |
| 5       | [1.300435]      | [1.310486]            | [0,010051]    | [Nenhum impacto significativo]     |

1° ping

- 1 0 1 ping 64 ----- 2 0.1 3.0 -1 344
- 1.0008 0 1 ping 64 ----- 2 0.1 3.0 -1 344 r 1.010851 0 1 ping 64 ----- 2 0.1 3.0 -1 344

2° ping

- 1.010851 1 3 ping 64 ----- 2 0.1 3.0 -1 344
- 1.010851 1 3 ping 64 ----- 2 0.1 3.0 -1 344 r 1.020902 1 3 ping 64 ----- 2 0.1 3.0 -1 344

3° ping

- 1.020902 3 1 ping 64 ----- 2 3.0 0.1 -1 364
- 1.020902 3 1 ping 64 ----- 2 3.0 0.1 -1 364 r 1.030954 3 1 ping 64 ----- 2 3.0 0.1 -1 364

4° ping

- 1.030954 1 0 ping 64 ----- 2 3.0 0.1 -1 364
- 1.030954 1 0 ping 64 ----- 2 3.0 0.1 -1 364 r 1.041005 1 0 ping 64 ----- 2 3.0 0.1 -1 364

## 5° ping

- 1.3 0 1 ping 64 ----- 2 0.1 3.0 -1 635
- 1.300435 0 1 ping 64 ----- 2 0.1 3.0 -1 635 r 1.310486 0 1 ping 64 ----- 2 0.1 3.0 -1 635

## 7.3. Perguntas para Refletir e Discutir

### 1. Qual aplicação (FTP ou Ping) é mais sensível à latência? Por quê?

- O ping é mais sensível, pelo atraso do retorno do ACK comparado aos vários pacotes FTP enviados. Se houver atraso na rede, isso é imediatamente visível no tempo de resposta do Ping.

### 2. Como o throughput do FTP foi afetado pela capacidade do link?

- Pois, a capacidade máxima do canal (10 Mbps) limitou o throughput máximo, representando aproximadamente 10% da capacidade total do link. Throughput quase 1 Mbps para a capacidade do Link de 10 Mbps, representando 10%.

### 3. Em um cenário de telecirurgia, qual seria a prioridade: alto throughput para o vídeo HD (Pablo) ou alta responsividade para os comandos do bisturi (Flash)? Justifique.

- O mais alarmante entre essas duas opções, vai depender do cenário da cirurgia e dos técnicos ali habilitados. Num primeiro momento os comandos do bisturi devem ser priorizados, pois devem vir de comandos precisos e delicados, deixando o vídeo com uma qualidade baixa. Porém, quando não estiver usando o bisturi (algo que exige profissionalismo), pode-se alterar para a alta qualidade de vídeo.

---

## 8. Parte V: Perda de Pacotes – O Preço da Imperfeição

**Contexto Teórico:** A perda de pacotes ocorre quando um pacote não chega ao destino. A tolerância a essa perda varia drasticamente entre as aplicações, como os dados vitais do paciente (Data).

### 8.1. Simulação de Perda de Pacotes no NS2

- Criei e executei o script `lab_perda.tcl`, ajustando a taxa de erro de bit (`rate_`) para diferentes valores (ex: 1e-2, 1e-5) no `ErrorModel`.

**Entrega:** O código `lab_perda.tcl` utilizado.

```
# [Conferir Arquivo lab_perda.tcl]
```

### 8.2. Análise da Perda de Pacotes no Arquivo de Trace (.tr)

- Analisei o arquivo `lab_perda.tr` para calcular a taxa de perda de pacotes UDP e observar o comportamento do TCP.

**Cálculos da Taxa de Perda de Pacotes UDP:**

| <b>rate_ Configurado<br/>(ErrorModel)</b> | <b>Pacotes UDP<br/>Enviados</b> | <b>Pacotes UDP<br/>Recebidos</b> | <b>Pacotes<br/>Perdidos</b> | <b>Taxa de Perda<br/>(%)</b> |
|---|---------------------------------|----------------------------------|-----------------------------|------------------------------|
| [Valor 1 (e.g., 1e-2)]                    | [1123]                          | [1120]                           | [3]                         | [0,267%]                     |
| [Valor 2 (e.g., 1e-5)]                    | [1123]                          | [1119]                           | [4]                         | [0,356%]                     |

Para UDP - São os pacotes CBR (constant bit rate)

Taxa de perda de pacotes UDP: (Enviados - Recebidos) / Enviados

### Descrição do Comportamento do TCP:

- [Descreva o que você observou no trace file para o TCP, mencionando eventos de retransmissão (R) e ACKs, e como ele se diferencia do UDP em termos de entrega final]

Vemos uma entrega constante de pacotes via constant bit rate (representando o protocolo UDP), sem se importar se o pacote chegou realmente ao destino, sua entrega é rápida, porém não segura, sem retransmissões ou ACKs de confirmação. Diferentemente do que vemos no TCP, temos uma entrega não tão rápida, porém confiável, até mesmo com o estabelecimento da comunicação (pacote inicial), vemos um controle no TCP como retransmissões e recuperações rápidas.

## 8.3. Perguntas para Refletir e Discutir

### 1. Qual protocolo (UDP ou TCP) é mais afetado pela perda de pacotes em termos de entrega final? Por quê?

- O protocolo TCP é mais afetado pela perda de pacotes. Pois, percebe a perda de pacotes e faz o reenvio deles. Esse processo de retransmissão pode afetar a performance geral levar a latência na rede.

### 2. Como a taxa de perda configurada no script (rate\_) se compara à taxa de perda observada para o UDP?

- A taxa de perda configurada no script (1e-2 ou 1e-5) é a probabilidade de erro por bit num pacote. Entretanto, podemos observar que a taxa de perda observada é impactada pela taxa de erro de bit. Mas, com uma taxa mais alta (1e-2) tivemos uma perda mais baixa.

### 3. Dê exemplos de aplicações que toleram alta perda de pacotes e aplicações que não toleram nenhuma perda.

- Aplicações que toleram perdas: DNS (resolução de nomes de domínio) e SNMP (monitoramento de dispositivos). A perda de pacotes para ambos é tolerável, visto que suas retransmissões são rápidas e não custosas.

Aplicações que não toleram perdas: Transmissão de dados bancários e controle de processos automatizados na produção de uma fábrica. Não toleram, pois são cruciais ao processo, mesmo havendo retransmissão de pacotes, pode ser que seja tarde demais e cause erros.

## 9. Parte VI: Consolidação e Perspectivas Futuras



## Síntese do Aprendizado

- [Escreva uma síntese dos principais aprendizados sobre a relação entre os parâmetros de QoS (latência, jitter, throughput, perda) e o desempenho de diferentes aplicações, utilizando os resultados dos experimentos. Faça um link com a **narrativa da telecirurgia** e proponha uma **solução baseada em QoS** para otimizar o desempenho das aplicações críticas nesse cenário desafiador (vídeo HD, comandos táteis, voz, dados do paciente).]

Latência é medida em milissegundos (ms), representa o tempo total para um pacote ir da origem ao destino. Uma alta latência afeta a responsividade, isso é, a capacidade do sistema de responder a comandos ou solicitações de forma rápida, como os comandos médicos. Vemos nesse laboratório que a latência aumenta conforme o atraso do link / conexão. Algo como uma telecirurgia a distância (Tóquio ao Brasil) teria um atraso de link alto.

| link_delay Configurado  | Timestamp Envio | Timestamp Recebimento | Latência Calculada |
|-------------------------|-----------------|-----------------------|--------------------|
| [Valor 1 (e.g., 10ms)]  | [0,69]          | [0,708]               | [0,018]            |
| [Valor 2 (e.g., 100ms)] | [0,85]          | [0,958]               | [0,108]            |
| [Valor 3 (e.g., 500ms)] | [0,77]          | [1,278]               | [0,508]            |

Jitter é a variação no atraso da entrega de pacotes, isso é, os pacotes chegam ao destino com atraso irregular, afetando o desempenho de aplicações em tempo real, como chamadas de voz (VoIP) e videoconferências, resultando em áudio cortado, vídeo com falhas ou congelado e atrasos na comunicação.

Vemos nesse laboratório que mesmo com uma chamada local básica do Google Meet, temos um Jitter e uma fração de perda maiores do que o aceitável. Logo, a telecirurgia teria graves problemas, como comandos por áudio terem ruídos ou serem incompletos, ou o vídeo ficar congelado na maioria do tempo.

- **Interarrival Jitter:** 1550922407 ms -> são quase 18 dias
- **Fraction Lost:**  $83 / 256 = 0,32422 \times 100 = 32,422\%$  (ou % se convertido)
- **Cumulative Number of Packets Lost:** 6701384

Throughput é medida em bits por segundo (bps), kilobits por segundo (Kbps) ou megabits por segundo (Mbps). Representa a capacidade efetiva de transferência de dados, considerando perdas, retransmissões e overhead de protocolos (quantidade real de dados).

Vemos nesse laboratório que o valor de Throughput chega perto de 1Mbps, que ao ser comparado com valores de outras redes, chega perto de um rede wi-fi 4G. Neste contexto, vemos a importância de uma boa infraestrutura das redes e seus processamentos intermediários. Pois a divisão pelo tempo é por parte impactada pelo processamento de dispositivos intermediários (roteadores, switches, servidores, NATs...), e a capacidade do link pode oferecer com que pacotes maiores sejam transportados por vez. Aumentando o Throughput da rede.

- Número de pacotes TCP recebidos: 3702

- Tamanho do pacote TCP (padrão NS2): 512 bytes (ou especifique se diferente) 1040
  - Tempo total da simulação para FTP (stop - start): [0.5 até 4.5 = 4] foram 4 segundos
  - Throughput = (Número de pacotes \* Tamanho do pacote) / Tempo
  - Throughput = (3702 \* 1040) / 4
  - Throughput (em Kbps/Mbps): 962520 = 962,52Kbps
- Tolerância a perdas é a capacidade de uma aplicação continuar funcionando mesmo que pacotes sejam perdidos durante a transmissão. Diferentes aplicações possuem níveis distintos de tolerância, determinando como devem ser priorizadas e protegidas pelos mecanismos de QoS.

Vemos nesse laboratório algo meio distinto, pois ao diminuirmos a taxa de erro por bit, a taxa de perda aumentou (deveria abaixar), nesse caso a perda aumenta em 1 pacote, sendo aceitável. Porém, considerando uma taxa de perda alta, teremos problemas na telecirurgia, pois podemos ter falhas nos comandos ou perda da qualidade de vídeo. Para tolerar essas perdas, deve-se ter retransmissões rápidas (Fast Retransmit - qualidade do TCP), redundâncias na rede e priorização de tráfego crítico. Claro, transmissões como a telecirurgia devem-se usar do protocolo UDP, por oferecer baixa latência, sem overhead de cabeçalhos. Porém, sem garantia de entrega.

Entende-se que unir uma característica TCP ao UDP, pode ser realizada utilizando protocolos como o QUIC/UDP criado pelo Google.

*Explicações sobre o protocolo QUIC/UDP fogem do escopo deste laboratório, porém cabe em explicações futuras*

| rate_ Configurado<br>(ErrorModel) | Pacotes UDP<br>Enviados | Pacotes UDP<br>Recebidos | Pacotes<br>Perdidos | Taxa de Perda<br>(%) |
|-----------------------------------|-------------------------|--------------------------|---------------------|----------------------|
| [Valor 1 (e.g., 1e-2)]            | [1123]                  | [1120]                   | [3]                 | [0,267%]             |
| [Valor 2 (e.g., 1e-5)]            | [1123]                  | [1119]                   | [4]                 | [0,356%]             |