Departamento de Informática e Matemática Aplicada – DIMAp

Programação com Socket UDP

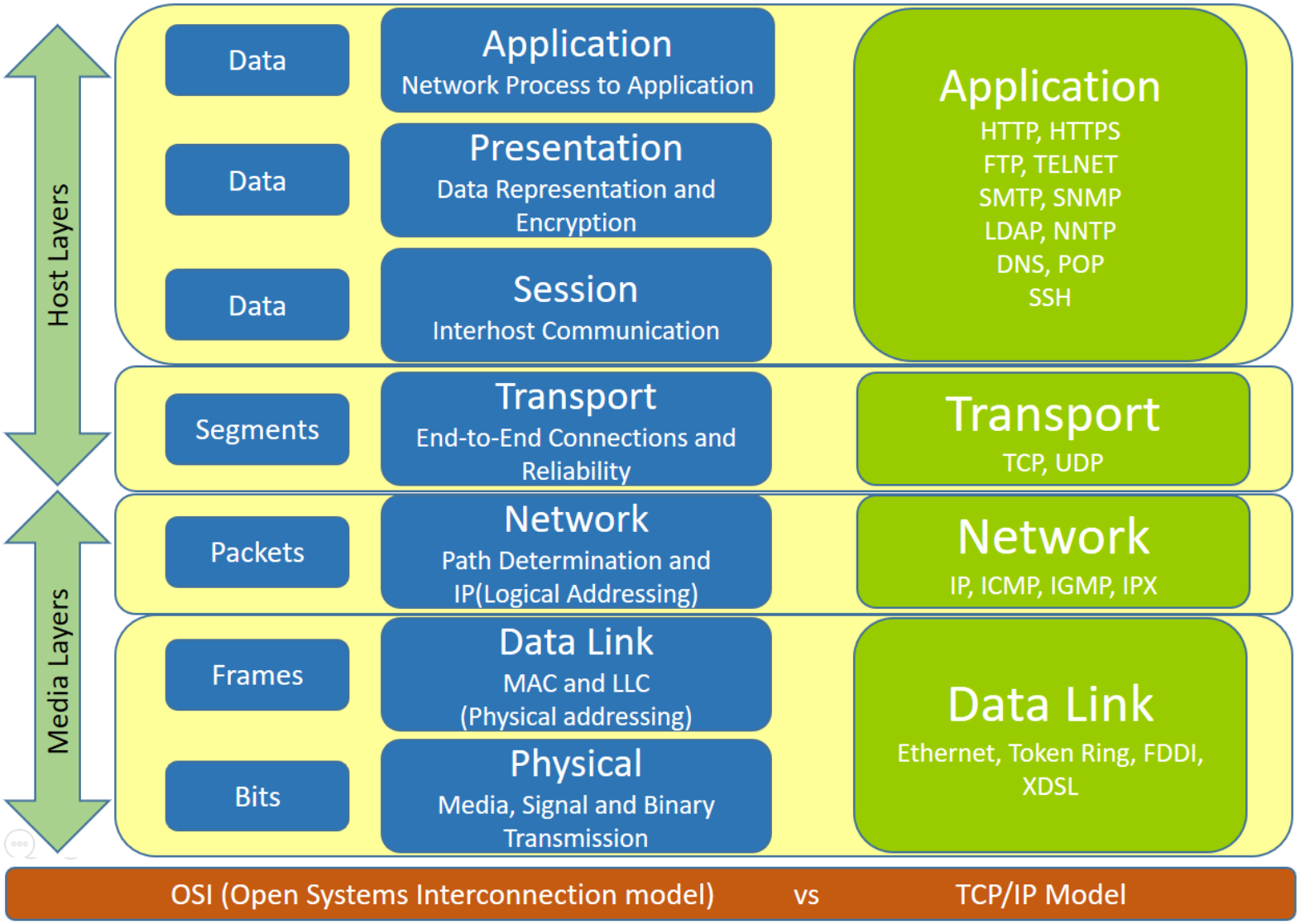
Prof. Nélio Cacho

Universidade Federal do Rio Grande do Norte
neliocacho@dimap.ufrn.br

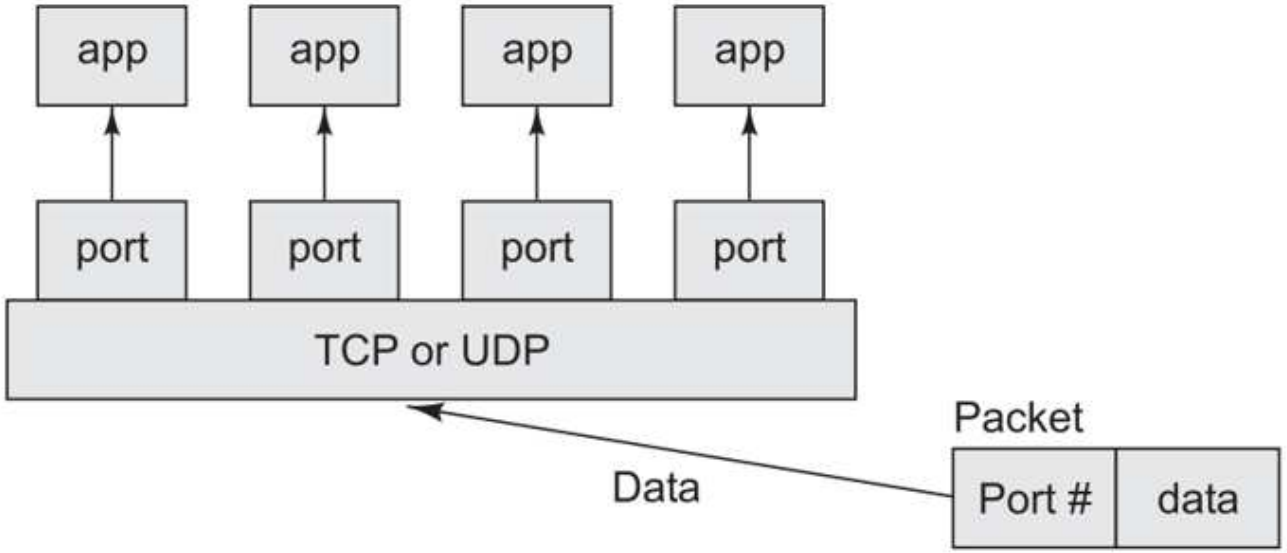
Hora de silenciar o celular...

- Manter o telefone celular sempre desligado/silencioso quando estiver em sala de aula;
- Nunca atender o celular na sala de aula.

Introdução


Host Layers
Media Layers
Data
Application
Network Process to Application
Data
Presentation
Data Representation and Encryption
Data
Session
Interhost Communication
Application
HTTP, HTTPS
FTP, TELNET
SMTP, SNMP
LDAP, NNTP
DNS, POP
SSH
Segments
Transport
End-to-End Connections and Reliability
Transport
TCP, UDP
Packets
Network
Path Determination and IP(Logical Addressing)
Network
IP, ICMP, IGMP, IPX
Frames
Data Link
MAC and LLC (Physical addressing)
Bits
Physical
Media, Signal and Binary Transmission
Data Link
Ethernet, Token Ring, FDDI, XDSL
OSI (Open Systems Interconnection model)
vs
TCP/IP Model

Endereçamento



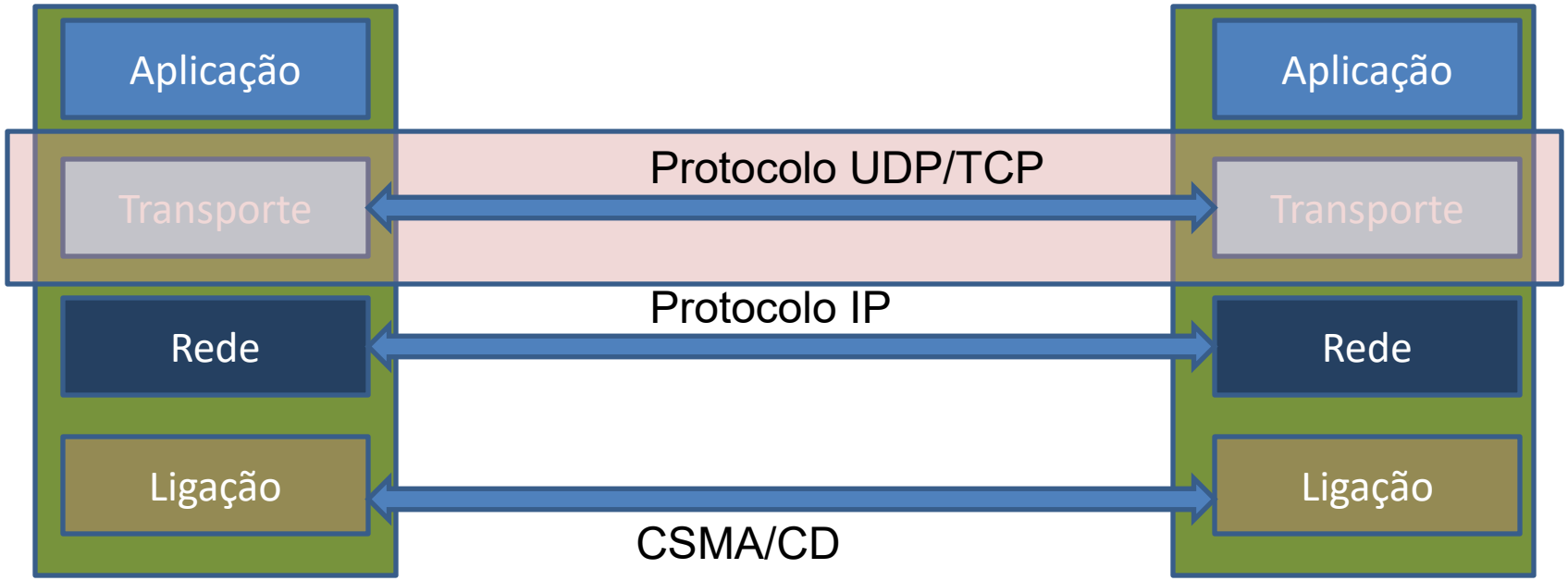
TCP/UDP packets to port/process mapping

Uma porta pode ser definida como um canal para transmitir os dados de entrada encaminhados para uma interface específica no computador host, utilizando os protocolos TCP ou UDP. Uma porta possui um número de 16 bits e varia de 0 a 65.535; dentro desse intervalo, as portas de 0 a 1.023 são reservadas para HTTP, FTP e outros serviços do sistema.

Endereçamento

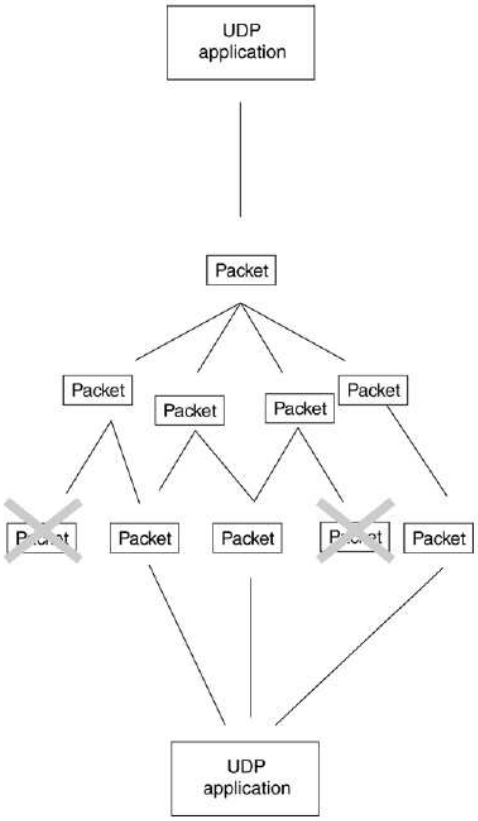
Port Number	Process Name	Protocol	Description
20	FTP-DATA	TCP	File transfer – data
21	FTP	TCP	File transfer – control
22	SSH	TCP	Secure shell
23	TELNET	TCP	Telnet
25	SMTP	TCP	Simple Mail Transfer Protocol
53	DNS	TCP and UDP	Domain Name System
69	TFTP	UDP	Trivial File Transfer Protocol
80	HTTP	TCP and UDP	Hypertext Transfer Protocol
110	POP3	TCP	Post Office Protocol 3
123	NTP	TCP	Network Time Protocol
143	IMAP	TCP	Internet Message Access Protocol
443	HTTPS	TCP	Secure implementation of HTTP

Como realizar a comunicação ?



Protocolo UDP

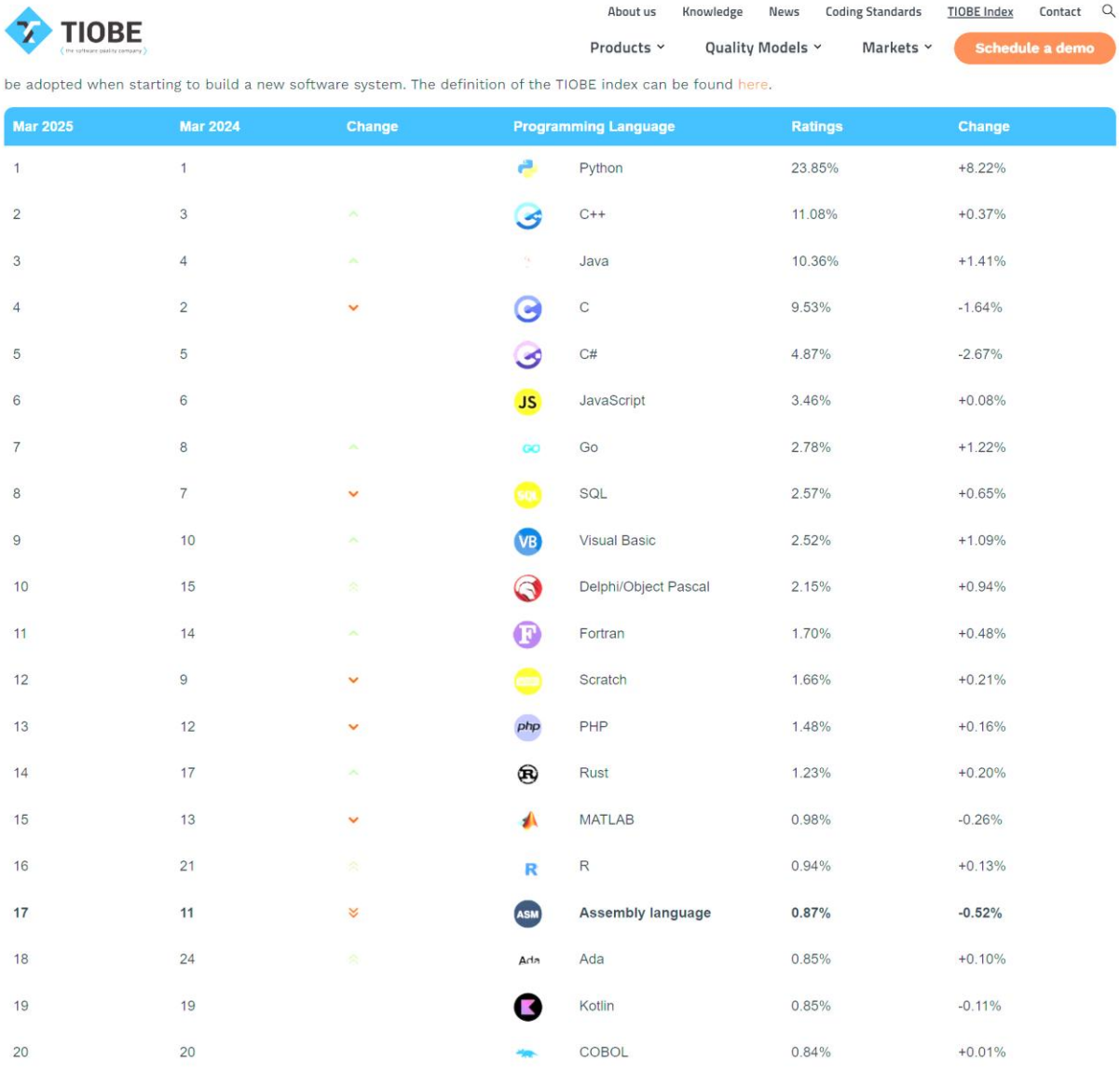
- UDP (User Datagram Protocol - protocolo de datagramas do utilizador) corresponde a um protocolo não orientado a conexão, ou seja, sem confiabilidade, já que não há garantia de envio/recebimento de pacotes.



Protocolo UDP

- UDP é utilizado pelos seguintes protocolos:
  - **Real Time Streaming Protocol (RTSP)**: Este protocolo é utilizado para controlar a transmissão de mídia.
  - **Routing Information Protocol (RIP)**: Este protocolo determina a rota usada para transmitir pacotes.
  - **Domain Name System (DNS)**: Este protocolo pesquisa o nome de domínio da Internet e retorna seu endereço IP.
  - **Network Time Protocol (NTP)**: Este protocolo sincroniza os relógios na Internet.
  - **HTTP 3.0:** A nova versão do HTTP é baseada no protocolo UDP.

Por que usar Java?



TIOBE

About us | Knowledge | News | Coding Standards | TIOBE Index | Contact

Products | Quality Models | Markets | Schedule a demo

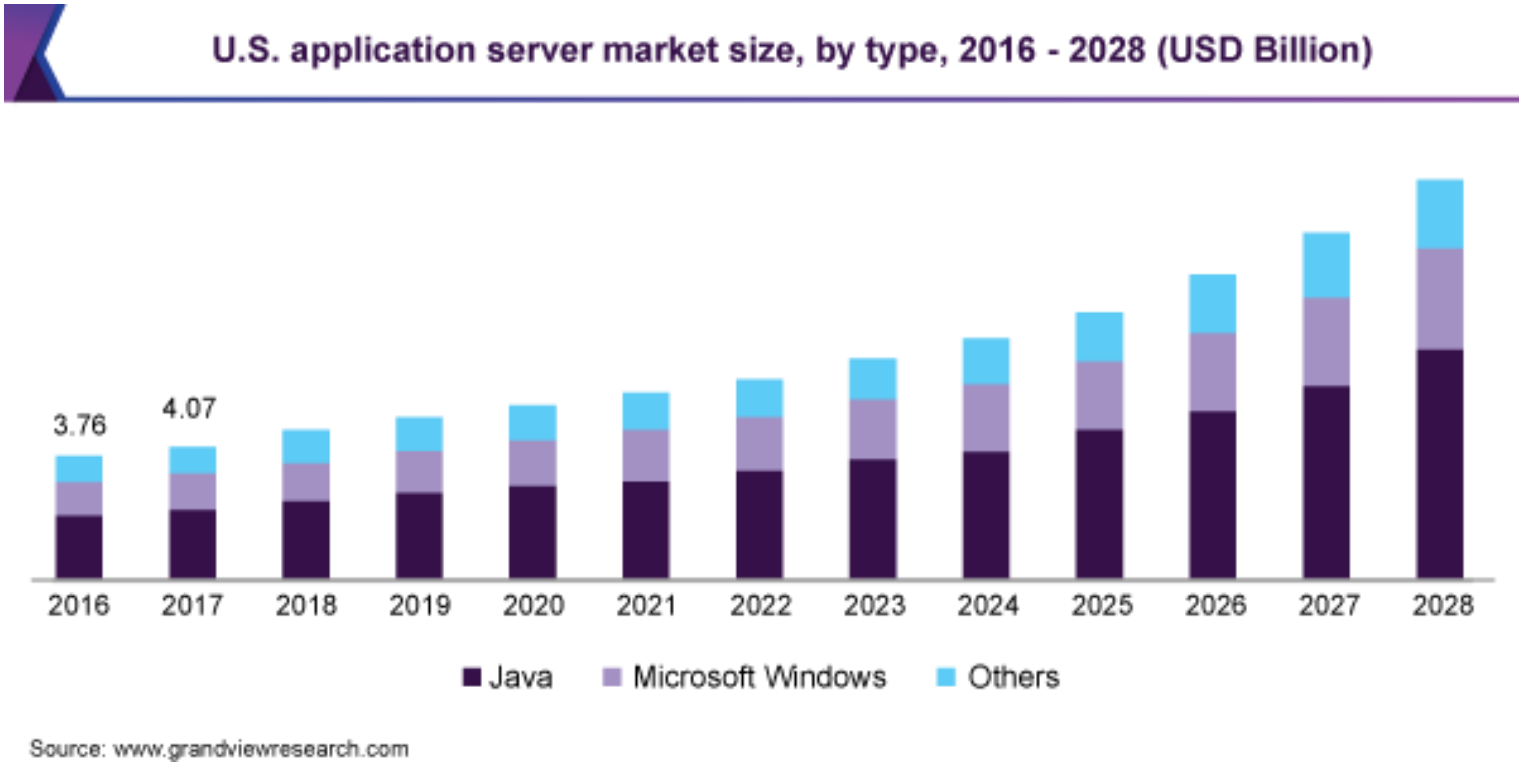
be adopted when starting to build a new software system. The definition of the TIOBE index can be found here.

Mar 2025	Mar 2024	Change	Programming Language	Ratings	Change
1	1		Python	23.85%	+8.22%
2	3	↑	C++	11.08%	+0.37%
3	4	↑	Java	10.36%	+1.41%
4	2	↓	C	9.53%	-1.64%
5	5		C#	4.87%	-2.67%
6	6		JavaScript	3.46%	+0.08%
7	8	↑	Go	2.78%	+1.22%
8	7	↓	SQL	2.57%	+0.65%
9	10	↑	Visual Basic	2.52%	+1.09%
10	15	⇈	Delphi/Object Pascal	2.15%	+0.94%
11	14	↑	Fortran	1.70%	+0.48%
12	9	↓	Scratch	1.66%	+0.21%
13	12	↓	PHP	1.48%	+0.16%
14	17	↑	Rust	1.23%	+0.20%
15	13	↓	MATLAB	0.98%	-0.26%
16	21	⇈	R	0.94%	+0.13%
17	**11**	⇊	**Assembly language**	**0.87%**	**-0.52%**
18	24	⇈	Ada	0.85%	+0.10%
19	19		Kotlin	0.85%	-0.11%
20	20		COBOL	0.84%	+0.01%

https://www.tiobe.com/tiobe-index/

Por que usar Java?

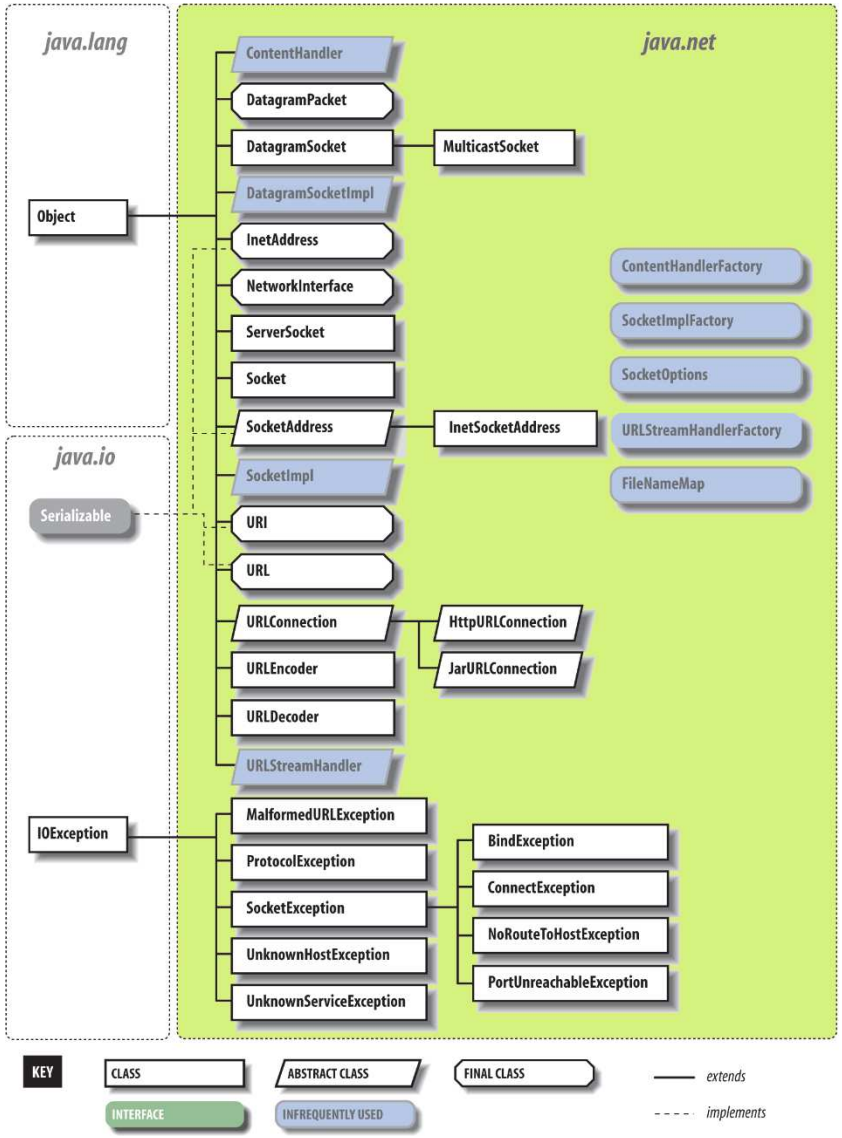
- Das 100 maiores empresas por receita, 96 utilizam Java nos seus sistemas.
- Exe.: Google, Amazon, Microsoft, IBM, Mastercard, Visa, PayPal, Alibaba, eBay, Walmart, Twitter, LinkedIn, Netflix, Spotify,Uber, Tesla, etc.



Package java.net

- Fornece as classes para a implementação de aplicações em rede
- Possui 28 classes, 6 interfaces e 11 exceções
- Serão estudadas apenas as classes principais

Package java.net



java.net.InetAddress

- Representa um endereço IP

- Pode ter 32 ou 128 bits
  - Veja as subclasses: Inet4Address e Inet6Address
- É usada pela maioria das classes Java que fazem conexão de rede
- Possui apenas dois campos privados:
  - String hostName
  - int address
- Não há construtores públicos

Inicializando InetAddress

- InetAddress address = InetAddress.getByName("www.imd.ufrn.br");

  System.out.println(address);

```
IP address : 177.20.147.222
```

- System.out.println("HostName: " + address.getHostName());

Exemplo (1) de uso do InetAddress

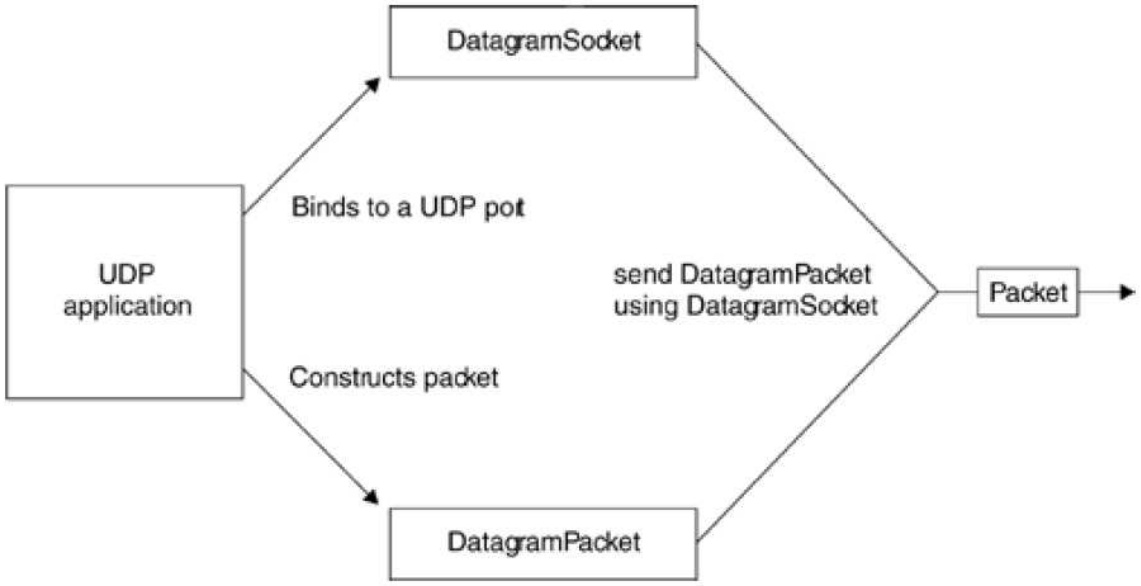
```
import java.net.*;

public class LocalHostDemo
{
    public static void main(String args[])
    {
        System.out.println ("Looking up local host");
        try
        {
              InetAddress localAddress = InetAddress.getLocalHost();

                    System.out.println ("IP address : " +  localAddress.getHostAddress() );
        }
        catch (UnknownHostException uhe)
        {
            System.out.println ("Error - unable to resolve localhost");
        }
    }
}
```

Exemplo (2) de uso do InetAddress

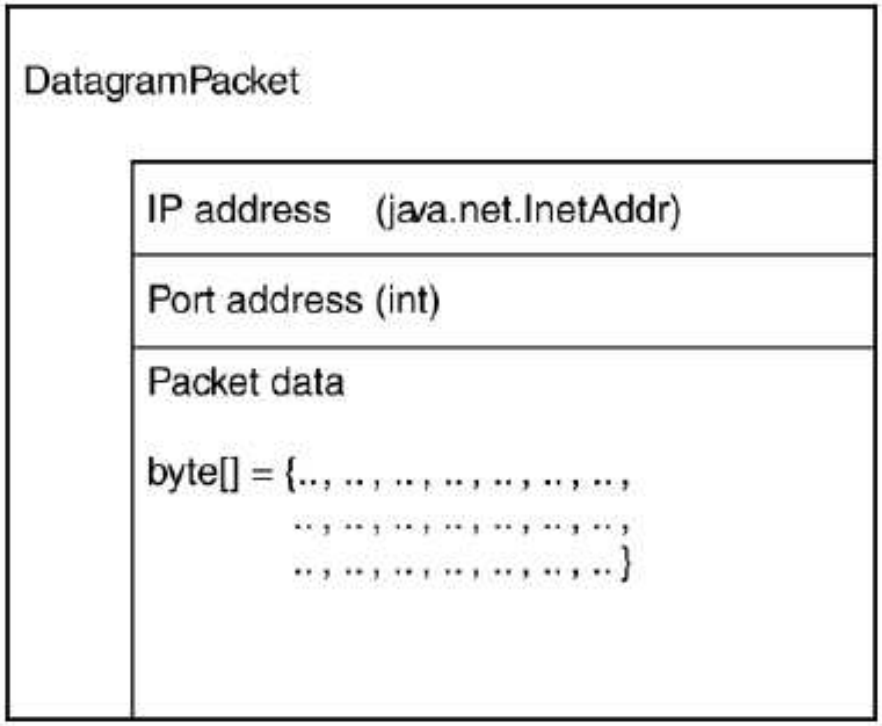
```
import java.net.*;

public class NetworkResolverDemo
{
    public static void main(String args[])
    {
        if (args.length != 1)
        {
            System.err.println ("Syntax - NetworkResolverDemo host");
            System.exit(0);
        }
       System.out.println ("Resolving " + args[0]);
       try
        {
          InetAddress addr = InetAddress.getByName  ( args[0] );

            System.out.println ("IP address : " +  addr.getHostAddress() );
            System.out.println ("Hostname : " +   addr.getHostName() );
        }
        catch (UnknownHostException uhe)
        {
            System.out.println ("Error - unable to resolve hostname" );
       }
       }
}
```

UDP em Java

- Java Suporta UDP através das classes:
  - Java.net.DatagramPacket
    - Representa a informação a ser enviada ou recebida
  - Java.net.DatagramSocket
    - Fornece os meios para enviar e receber informações (DatagramPacket)



Estrutura do DatagramPacket



Modificando DatagramPacket

DatagramPacket sendPacket = **new DatagramPacket(barray,barray.length)*;***

. . .

sendPacket.setPort(5000);

Criando DatagramSocket

DatagramSocket: Creation

```
DatagramSocket()
DatagramSocket(int localPort)
DatagramSocket(int localPort, InetAddress localAddr)
```

DatagramSocket socket_server = **new DatagramSocket(*4050);***

DatagramSocket socket_client = **new DatagramSocket(*); //usa porta disponível***

Enviando e Recebendo ...

DatagramSocket: Sending and receiving

```
void send(DatagramPacket packet)
void receive(DatagramPacket packet)
```

socket.send(packet);

socket.receive(packet); // fica esperando até chegada do pacote

Definindo o Timeout do socket

DatagramSocket: Options

```
int getSoTimeout()
void setSoTimeout(int timeoutMillis)
```

- Obtém e determina, respectivamente, o tempo máximo que o método ***receive*** irá ficar bloqueado. Se o tempo expirar, um exceção *InterruptedIOException* é lançada.

Exemplo Servidor UDP

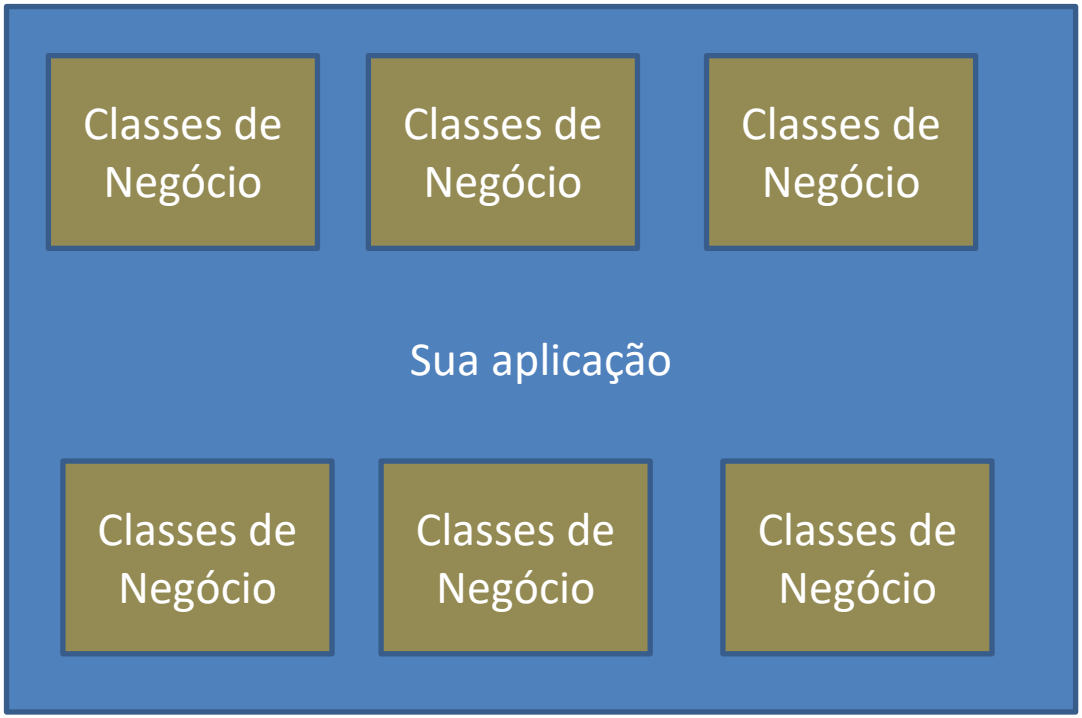
```
public class UDPServer {
   public UDPServer() {
      System.out.println("UDP Server Started");
      try {
DatagramSocket serverSocket = new DatagramSocket(9003);
while (true) {
         byte[] receiveMessage = new byte[1024];
         DatagramPacket receivePacket = new DatagramPacket(
            receiveMessage, receiveMessage.length);
         serverSocket.receive(receivePacket);
         String message = new String(receivePacket.getData());
         System.out.println("Received from client: " + message
               + "\nFrom: " + receivePacket.getAddress());
        }
} catch (IOException e) {e.printStackTrace();}
      System.out.println("UDP Server Terminating");
   }
   public static void main(String[] args) {     new UDPServer();    }
}
```

Exemplo Cliente UDP

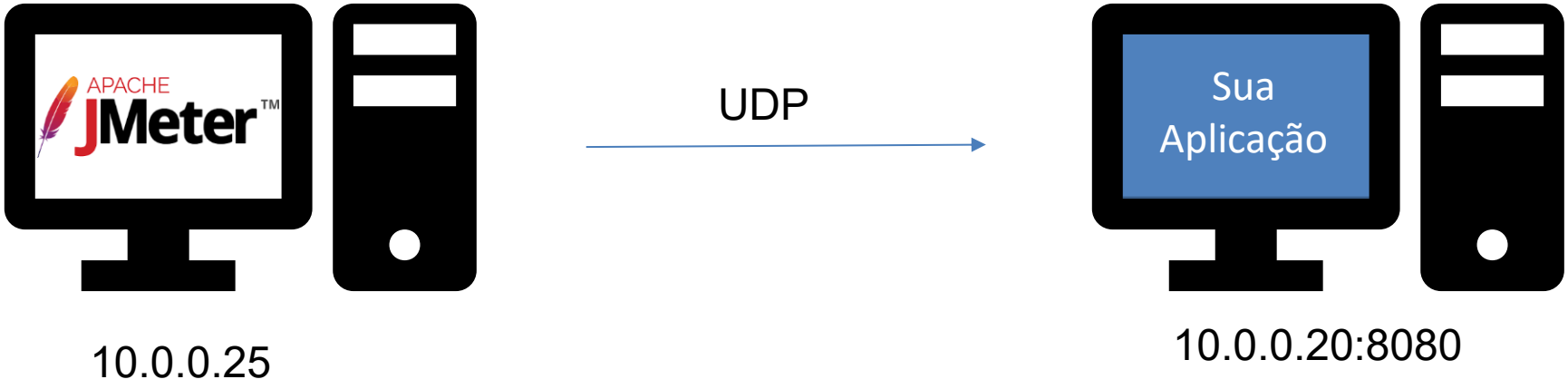
```
class UDPClient {
   public UDPClient() {
      System.out.println("UDP Client Started");
       Scanner scanner = new Scanner(System.in);
      try {
      DatagramSocket clientSocket = new DatagramSocket();
      InetAddress inetAddress = InetAddress.getByName("localhost");
            byte[] sendMessage;
             while (true) {
               System.out.print("Enter a message: ");
               String message = scanner.nextLine();
               if ("quit".equalsIgnoreCase(message)) {
                   break;
               }
               sendMessage = message.getBytes();
               DatagramPacket sendPacket = new DatagramPacket(
                  sendMessage, sendMessage.length,
                  inetAddress, 9003);
               clientSocket.send(sendPacket);
            }
         clientSocket.close();
      } catch (IOException ex) {      }
      System.out.println("UDP Client Terminating ");
   }
   public static void main(String args[]) {       new UDPClient();     } }
```

Ambiente Local

Modelagem da Aplicação



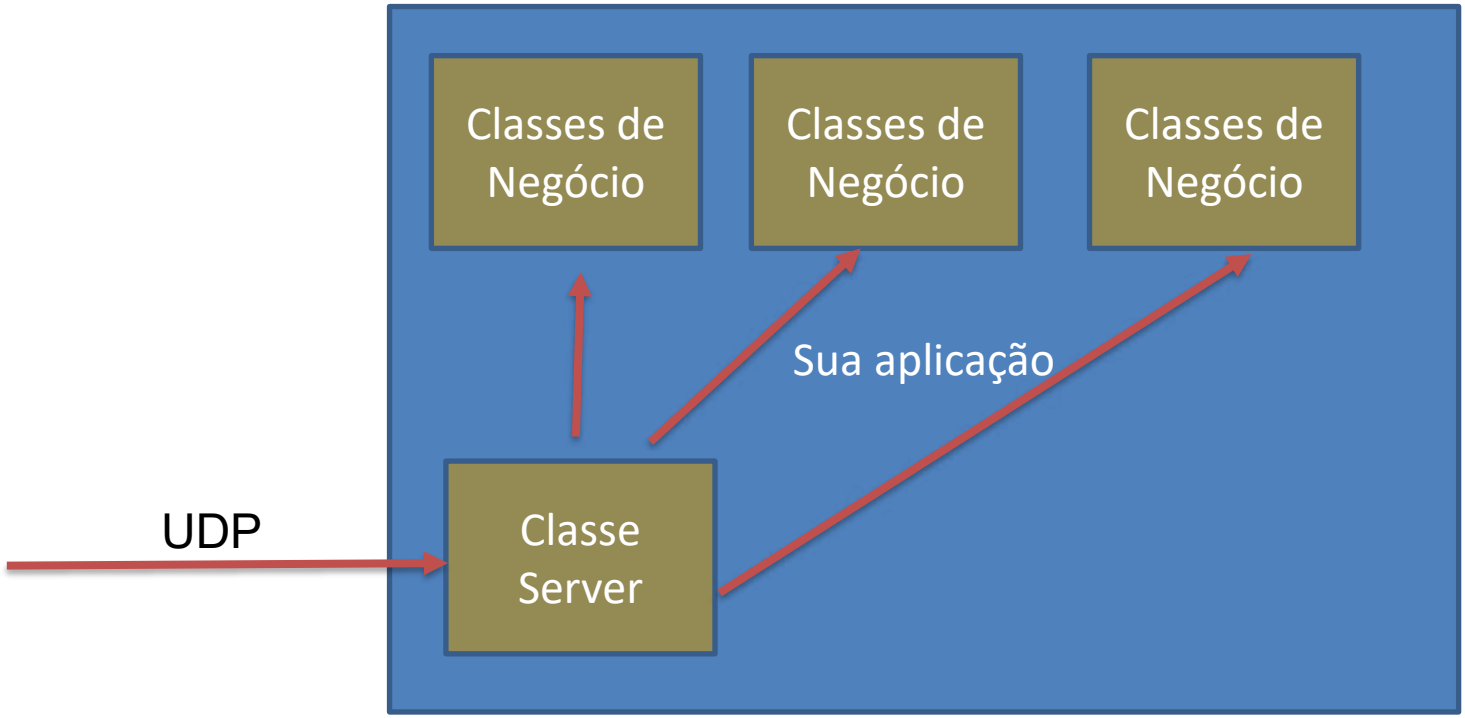
Como fazer para disponibilizar sua aplicação na Rede?



Classe de Negócio

```
public class Banco {
    private HashMap<Integer,Integer> contas = new HashMap<>();
    public void addConta(int numconta) {
        contas.put(numconta, 0);
    }
    public void depositar(int numconta, int valor) {
        int atual = contas.get(numconta);
        contas.put(numconta, atual+valor);
    }
    public int saldo(int numconta) {
        return contas.get(numconta);
    }
}
```

Como fazer para disponibilizar sua aplicação na Rede?



```java
public class UDPServerBanco {
    private Banco banco;
    public UDPServerBanco(String port) {
        banco = new Banco();
        String operacao =null;
        int conta = 0;
        int valor = 0;
        System.out.println("UDP Server Bank started");
        try {
            DatagramSocket serversocket = new DatagramSocket(Integer.parseInt(port));
            String resultadoOp;
            while (true) {
                byte[] receivemessage = new byte[1024];
                DatagramPacket receivepacket = new DatagramPacket(receivemessage, receivemessage.length);
                serversocket.receive(receivepacket);
                String message = new String(receivepacket.getData());
                resultadoOp=message;
                StringTokenizer tokenizer = new StringTokenizer(message, ";");
                while (tokenizer.hasMoreElements()) {
                    operacao = tokenizer.nextToken();
                    conta = Integer.parseInt(tokenizer.nextToken());
                    valor = Integer.parseInt(tokenizer.nextToken().trim());
                }
                switch (operacao) {
                    case "criar":
                        banco.addConta(conta);
                        break;
                    case "depositar":
                        banco.depositar(conta, valor);
                        break;
                    case "saldo":
                        resultadoOp = "R$"+banco.saldo(conta);
                        break;
                    }
                System.out.println("Operacao realizada:"+operacao+ "-"+ banco.saldo(conta) +"-"+ receivepacket.getAddress());
                String reply = "Confirmo Recebimento de:"+resultadoOp;
                byte[] replymsg = reply.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(replymsg,replymsg.length,
                        receivepacket.getAddress(),receivepacket.getPort());
                serversocket.send(sendPacket);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        System.out.println("UDP Bank server terminating");
    }
    public static void main(String[] args) {
        new UDPServerBanco(args[0]);
    }
}
```

```java
class UDPClientBanco {
    public UDPClientBanco() {
        System.out.println("UDP Client Bank Started");
        Scanner scanner = new Scanner(System.in);
        try {
            DatagramSocket clientSocket = new DatagramSocket();
            DatagramPacket sendPacket;
            InetAddress inetAddress = InetAddress.getByName("localhost");

            byte[] sendMessage;
            byte[] receivemessage = new byte[1024];

            //Enviar solicitação para abrir conta
            String message = "criar;1;0";
            sendMessage = message.getBytes();
            sendPacket = new DatagramPacket(
                        sendMessage, sendMessage.length,
                        inetAddress, 9004);
            clientSocket.send(sendPacket);

            //Receber resposta do Servidor - Conta Aberta
            DatagramPacket receivepacket = new DatagramPacket(receivemessage, receivemessage.length);
            clientSocket.receive(receivepacket);
            message = new String(receivepacket.getData());
            System.out.println(message);

            //Enviar solicitação para depositar
            message = "depositar;1;1000";
            sendMessage = message.getBytes();
            sendPacket = new DatagramPacket(
                            sendMessage, sendMessage.length,
                            inetAddress, 9004);
            clientSocket.send(sendPacket);

            //Receber resposta do Servidor - Deposito realizado
            Arrays.fill(receivemessage, (byte)0);
            receivepacket = new DatagramPacket(receivemessage, receivemessage.length);
            clientSocket.receive(receivepacket);
            message = new String(receivepacket.getData());
            System.out.println(message);

            //Enviar solicitação para obter saldo
            message = "saldo;1;0";
            sendMessage = message.getBytes();
            sendPacket = new DatagramPacket(
                            sendMessage, sendMessage.length,
                            inetAddress, 9004);
            clientSocket.send(sendPacket);

            //Receber resposta do Servidor - Saldo obtido
            Arrays.fill(receivemessage, (byte)0);
            clientSocket.receive(receivepacket);
            message = new String(receivepacket.getData());
            System.out.println(message);
            clientSocket.close();
        } catch (IOException ex) {
        }
        System.out.println("UDP Client Terminating ");
    }
    public static void main(String args[]) {
        new UDPClientBanco();
    }
}
```

Testando Aplicações Distribuídas

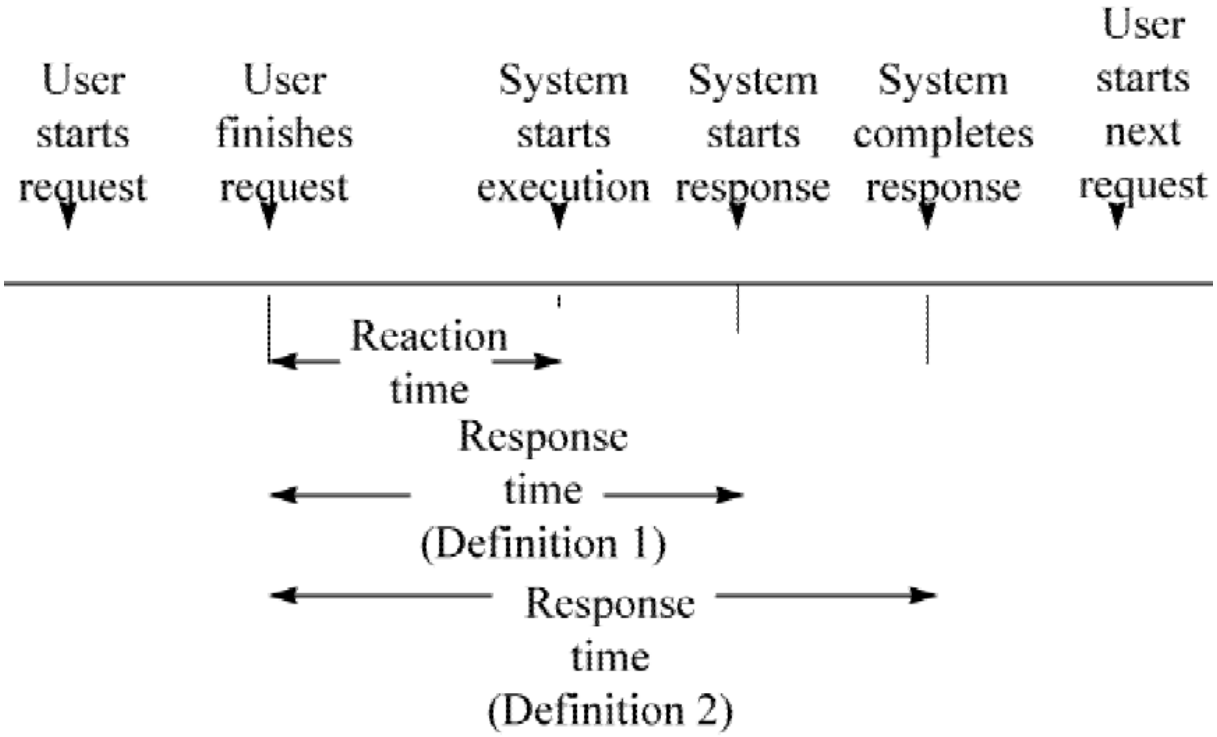
1. Teste de Carga: é uma avaliação onde é iniciado com uma carga baixa e vai aumentando gradativamente para descobrir a **capacidade máxima do sistema**.
2. Teste de Estresse: é uma avaliação em que é submetido a capacidade máxima do sistema de forma abrupta.
3. Teste de Desempenho: é uma avaliação realizada com uma carga constante e mantido por horas, analisando o **tempo de resposta** do sistema a cada interação do usuário.

O que testar?

Funcionalidades que têm muito **acesso simultâneo** (por exemplo, em um e-commerce, o cenário de compra é o mais importante e o que devemos garantir que tenha performance excelente) e funcionalidades em que a **requisição possa ser mais lenta**, como upload e download de arquivo.

Definições

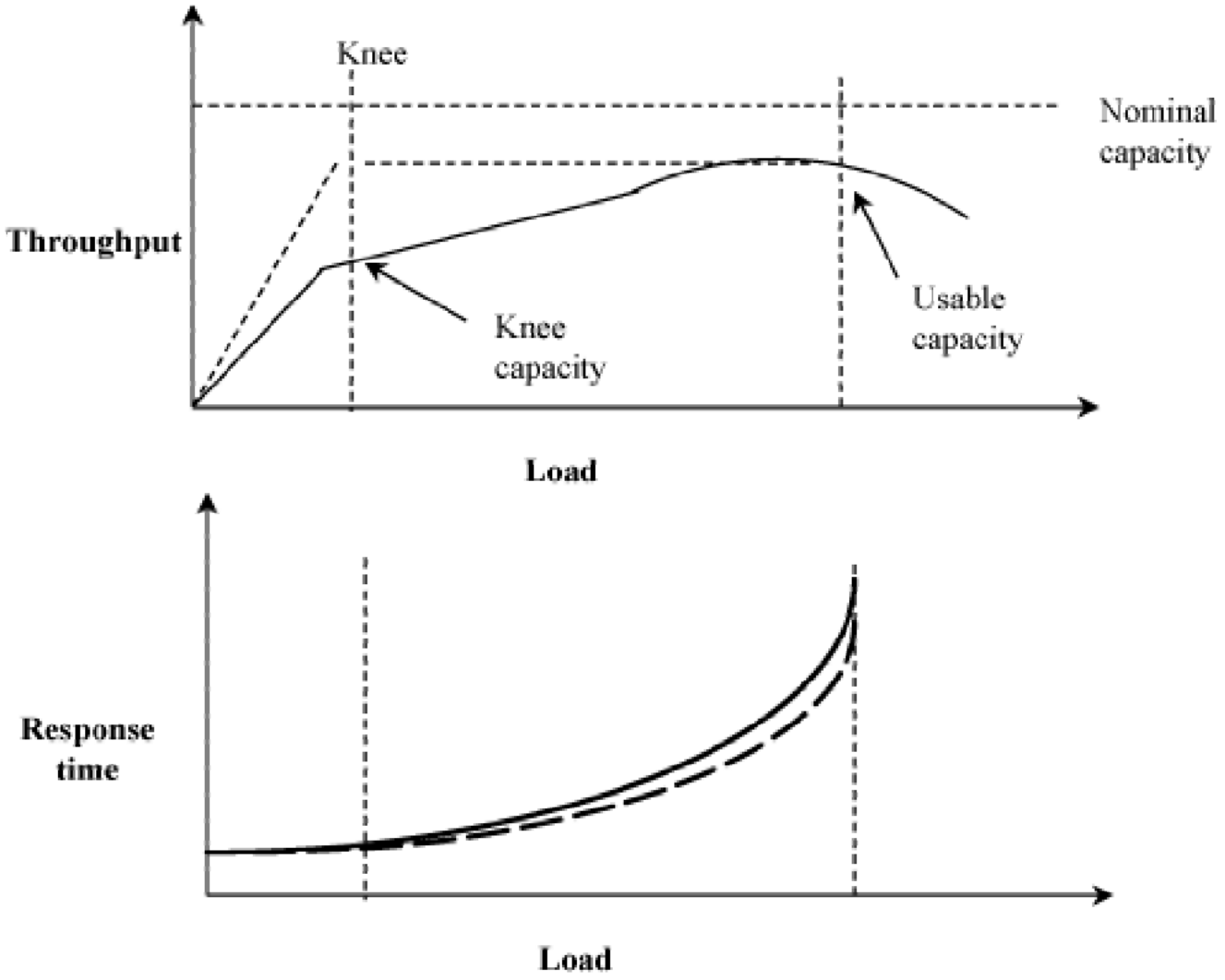
- **Response Time**: definido como intervalo entre a requisição do cliente e a resposta do servidor



Definições

- **Throughput**: número de requisições por unidade de tempo que o servidor consegue responder.

Analisando os resultados



JMeter

- Criada em 2007 pela Apache Software.
- Ferramenta mais utilizada para teste de performance.
- Possui código gratuito e aberto.
- Principais recursos: requisições, temporizador, extrator de expressão regular, asserções, utilização de variáveis, configuração e execução do teste e análise dos resultados.

Suporta testes com os tipos de serviço:

- Web - HTTP, HTTPS (Java, NodeJS, PHP, ASP.NET, …)
- SOAP / REST Webservices
- FTP
- Database via JDBC
- LDAP
- Message-oriented middleware (MOM) via JMS
- Mail - SMTP(S), POP3(S) and IMAP(S)
- Native commands or shell scripts
- TCP/UDP
- Java Objects

Preparação - Instalação JMeter

Instalação (Binaries ZIP):
http://jmeter.apache.org/download_jmeter.cgi

Dentro do bin:

- **Unix/Linux: ./jmeter.sh**
- **Windows: jmeter.bat**

É necessário configurar JAVA_HOME

Windows:

For illustrative purposes, we assume you have installed Java JDK at C:\tools\jdk:

1. Go to Control Panel.
2. Click on System.
3. Click on Advance System settings.
4. Add the Environment variable as follows:
   - ° Value: JAVA_HOME
   - ° Path: C:\tools\jdk

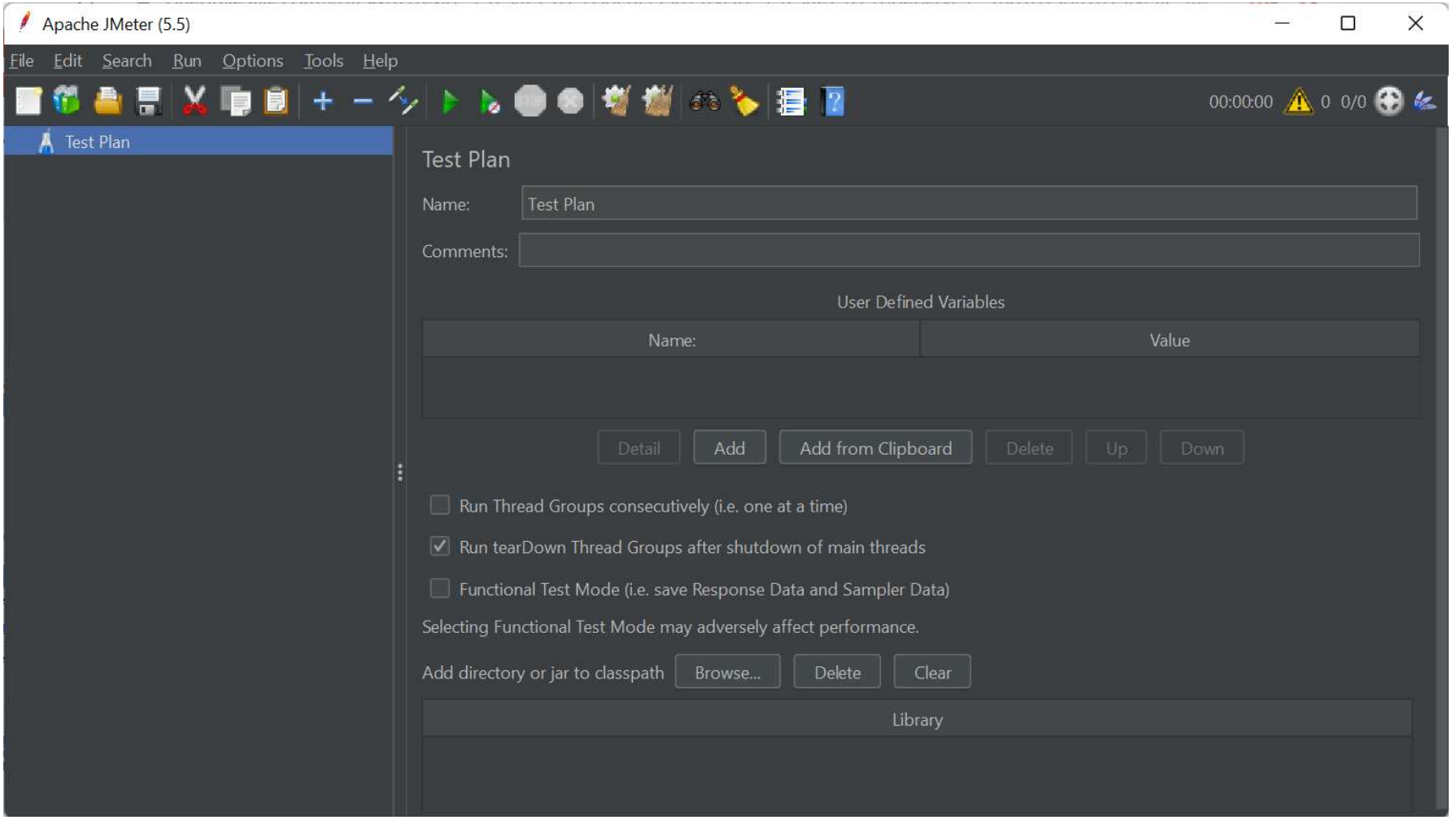
5. Locate Path (under System variables; bottom half of the screen).
6. Click on Edit.
7. Append %JAVA_HOME%/bin to the end of the existing path value (if any).

Unix:

For illustrative purposes, we assume you have installed Java JDK at /opt/tools/jdk:

1. Open a terminal window.
2. Export JAVA_HOME=/opt/tools/jdk.
3. Export PATH=$PATH:$JAVA_HOME.

JMeter tela de Entrada



Instalar Plugins no JMeter

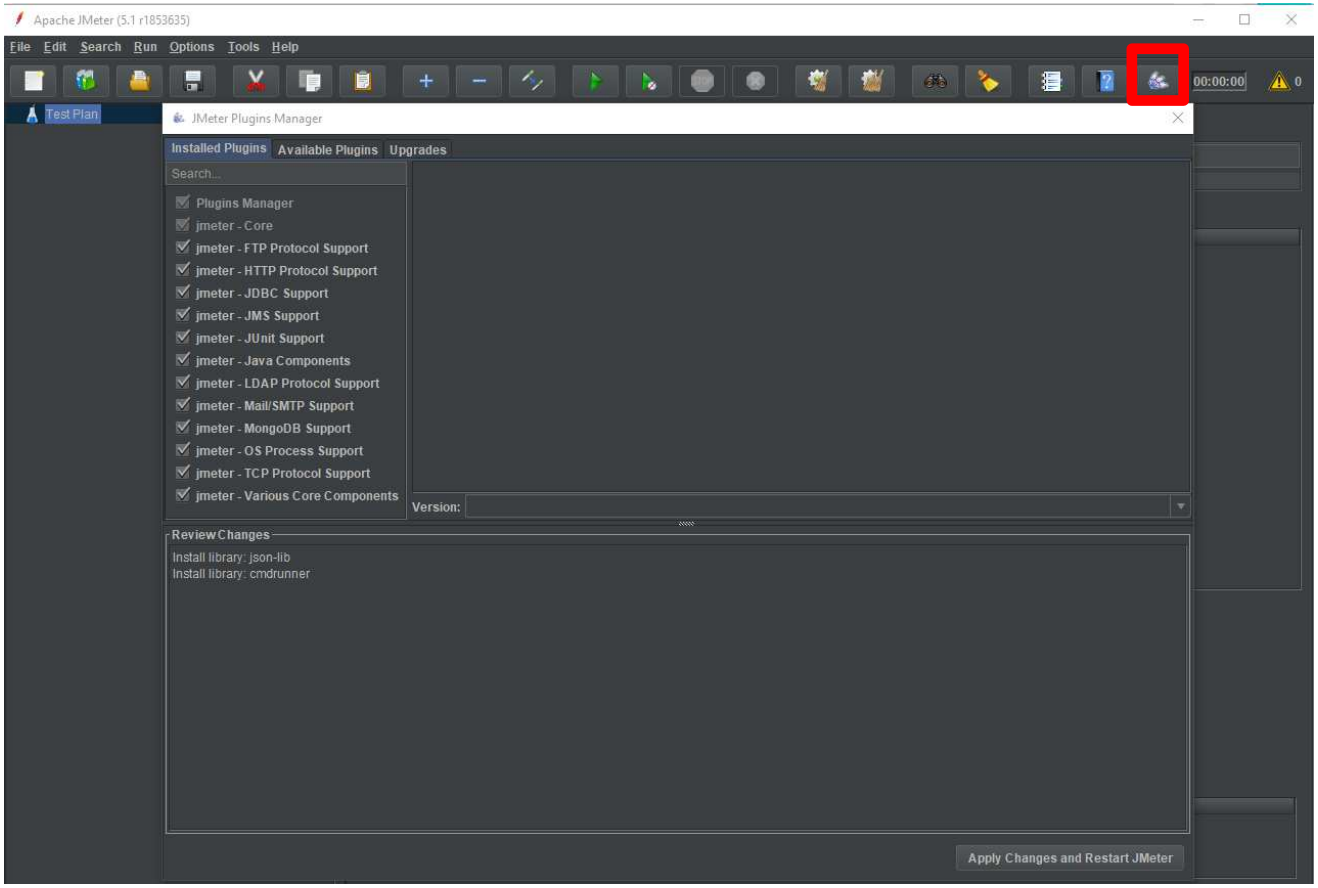
Instalação Plugin Manager

https://jmeter-plugins.org/wiki/PluginsManager/

Colocar dentro do lib/ext:

É necessário reiniciar JMeter

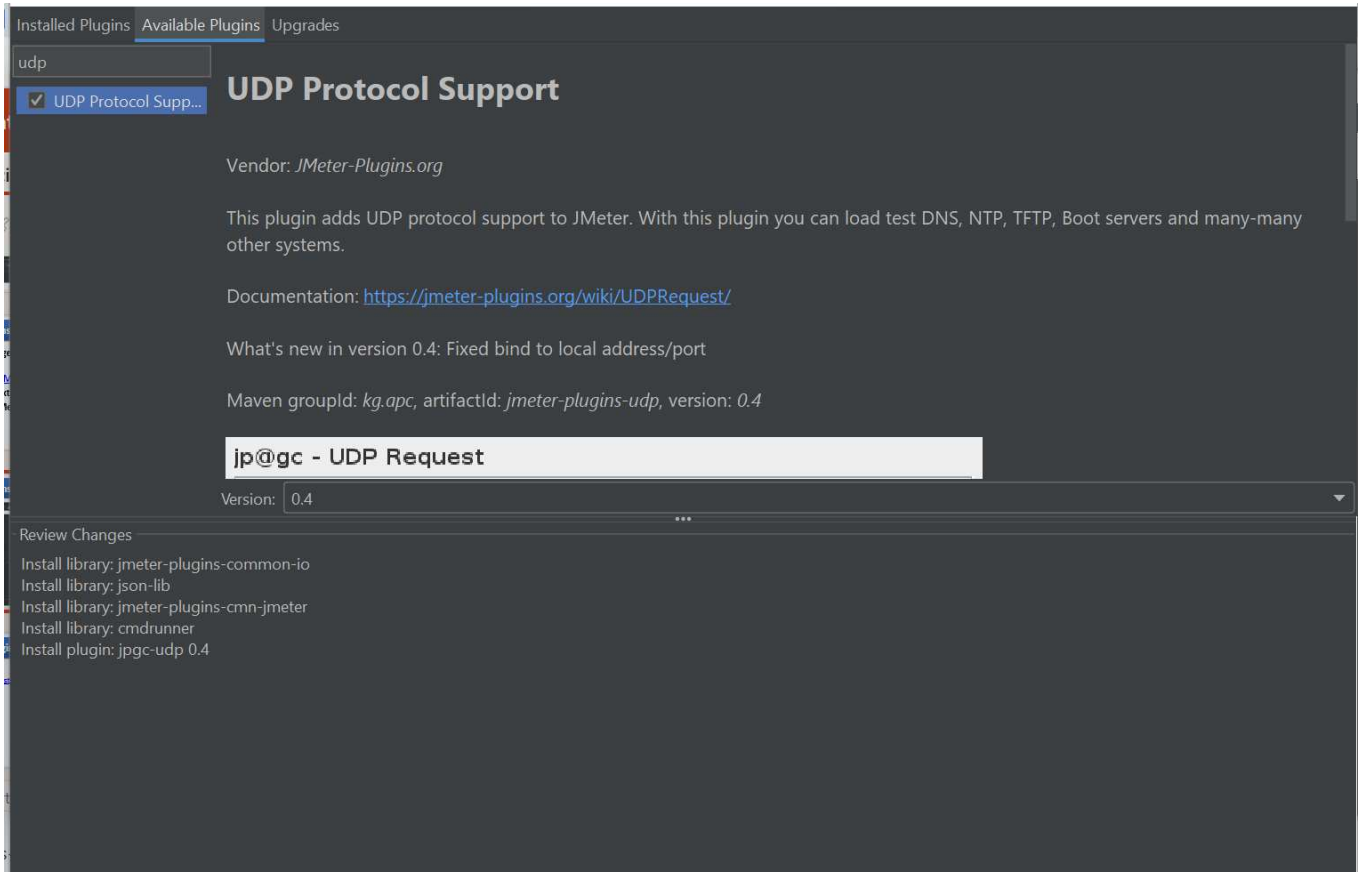
Selecionar Plugins para Instalar



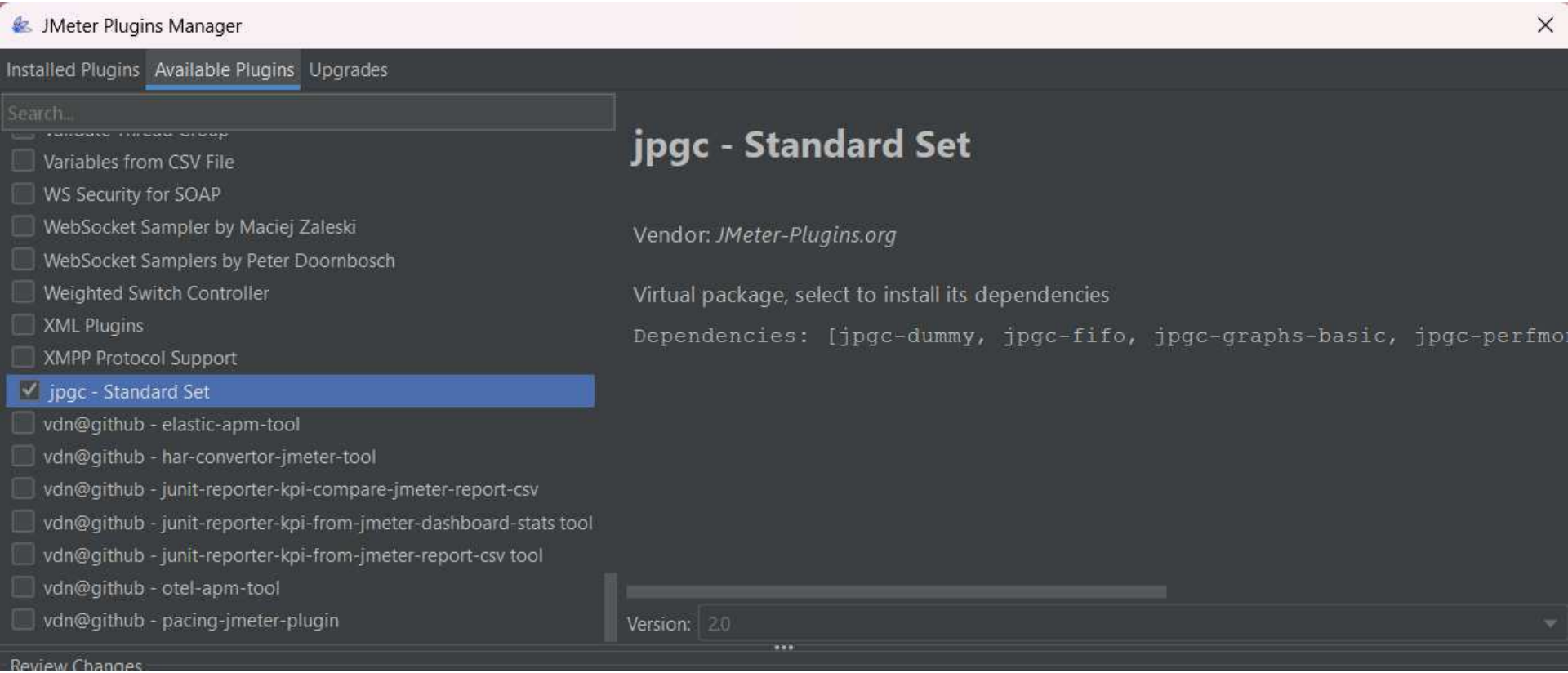
Instalar Plugin do UDP

- Instalar via https://jmeter-plugins.org/wiki/UDPRequest/
- Ou via Plugin Manager

Selecionar Plugins para Instalar



Selecionar Plugins para Instalar



Organização dos Testes no JMeter

- Plano de teste agrupa um ou mais Grupos de usuários.
- Grupo de Usuário(Thread Groups) agrupa um ou mais casos de teste
- Casos de teste agrupa um ou mais testes e funcionalidades

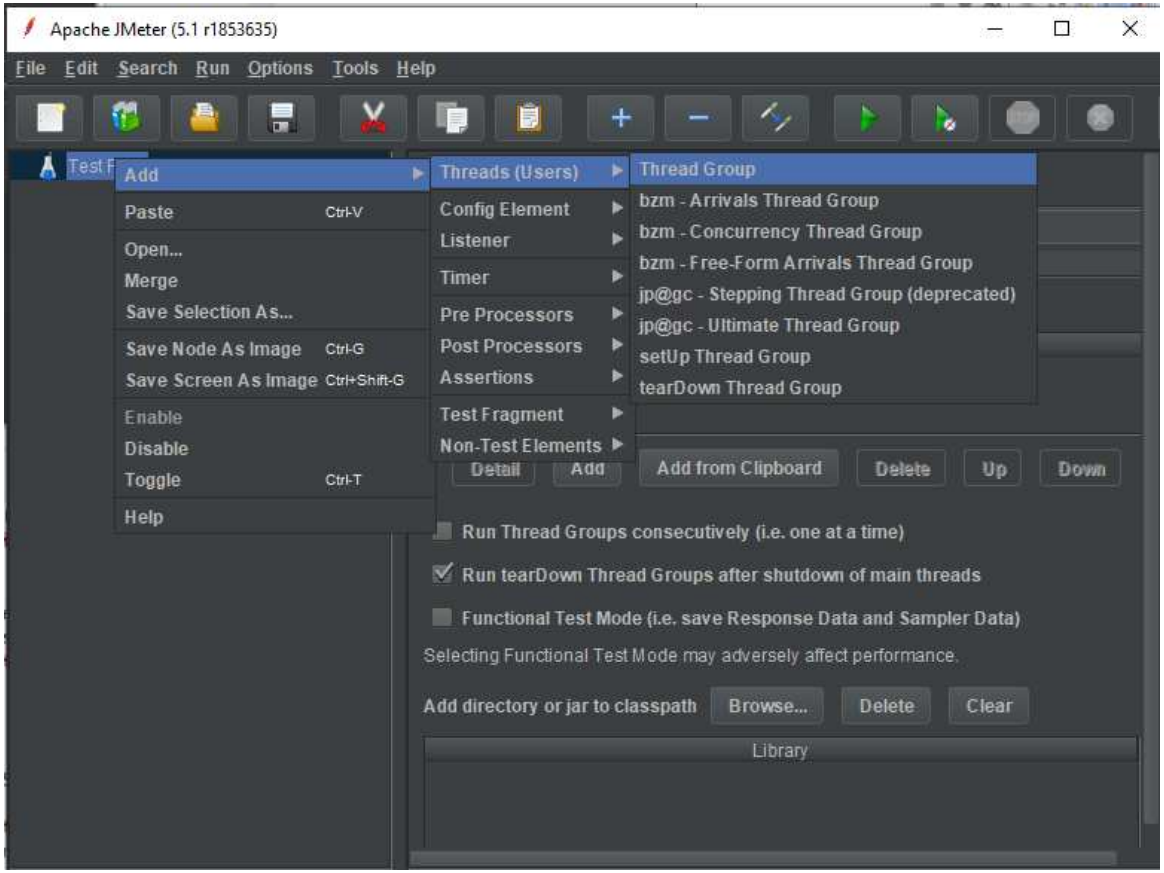
Grupos de usuários (Thread Groups)

Componente que controla a execução dos cenários, nele é definido:

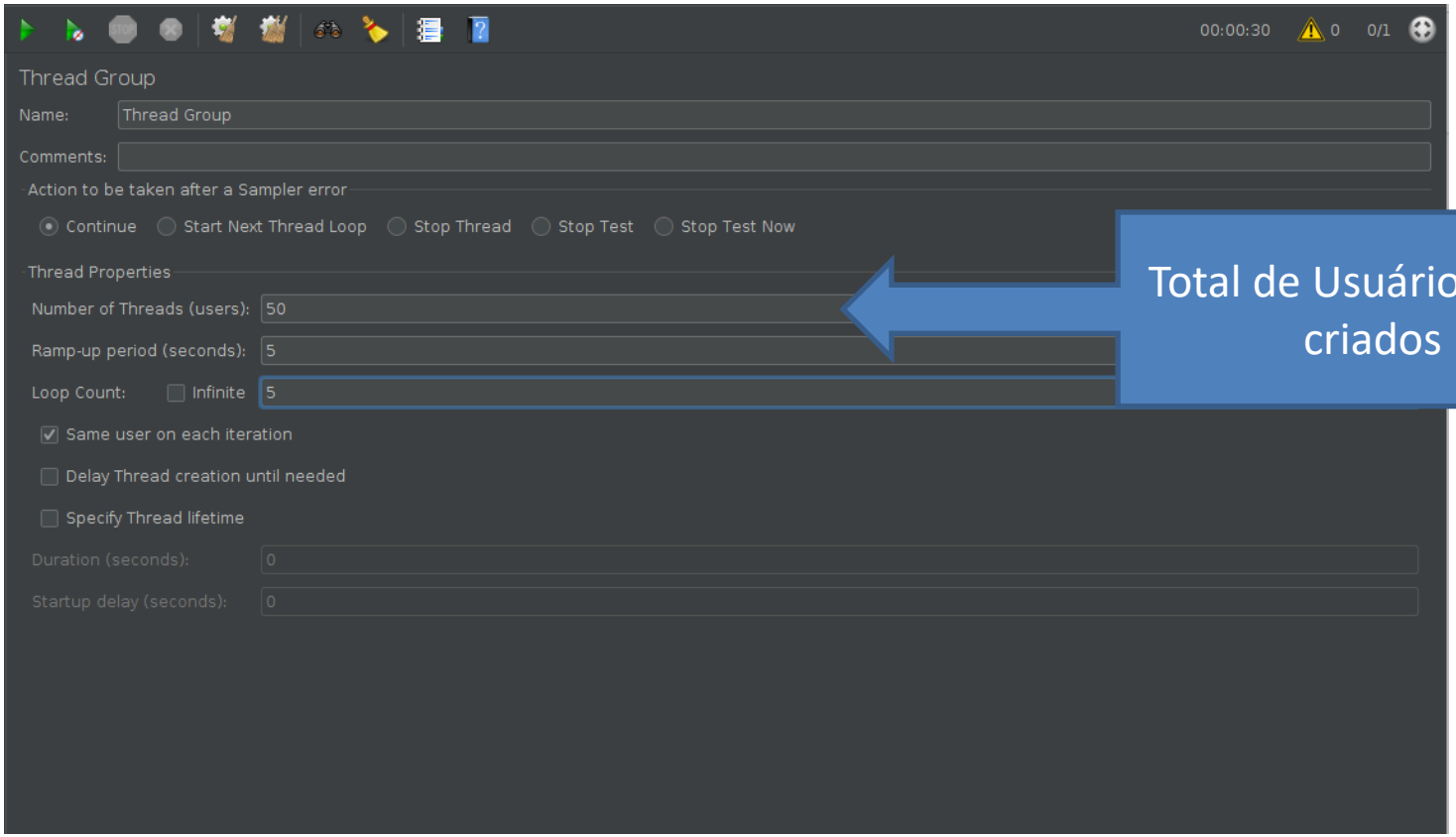
- Quantidade de usuários (threads) devem interagir com o sistema simultaneamente.
- Tempo de duração do teste/quantidade de execuções.
- Ação tomada pelo JMeter quando ocorrer um erro com um usuário.

Sendo, um grupo de usuário para cada caso de teste, podendo ser executados simultaneamente ou não.

Criar Grupo de Usuários

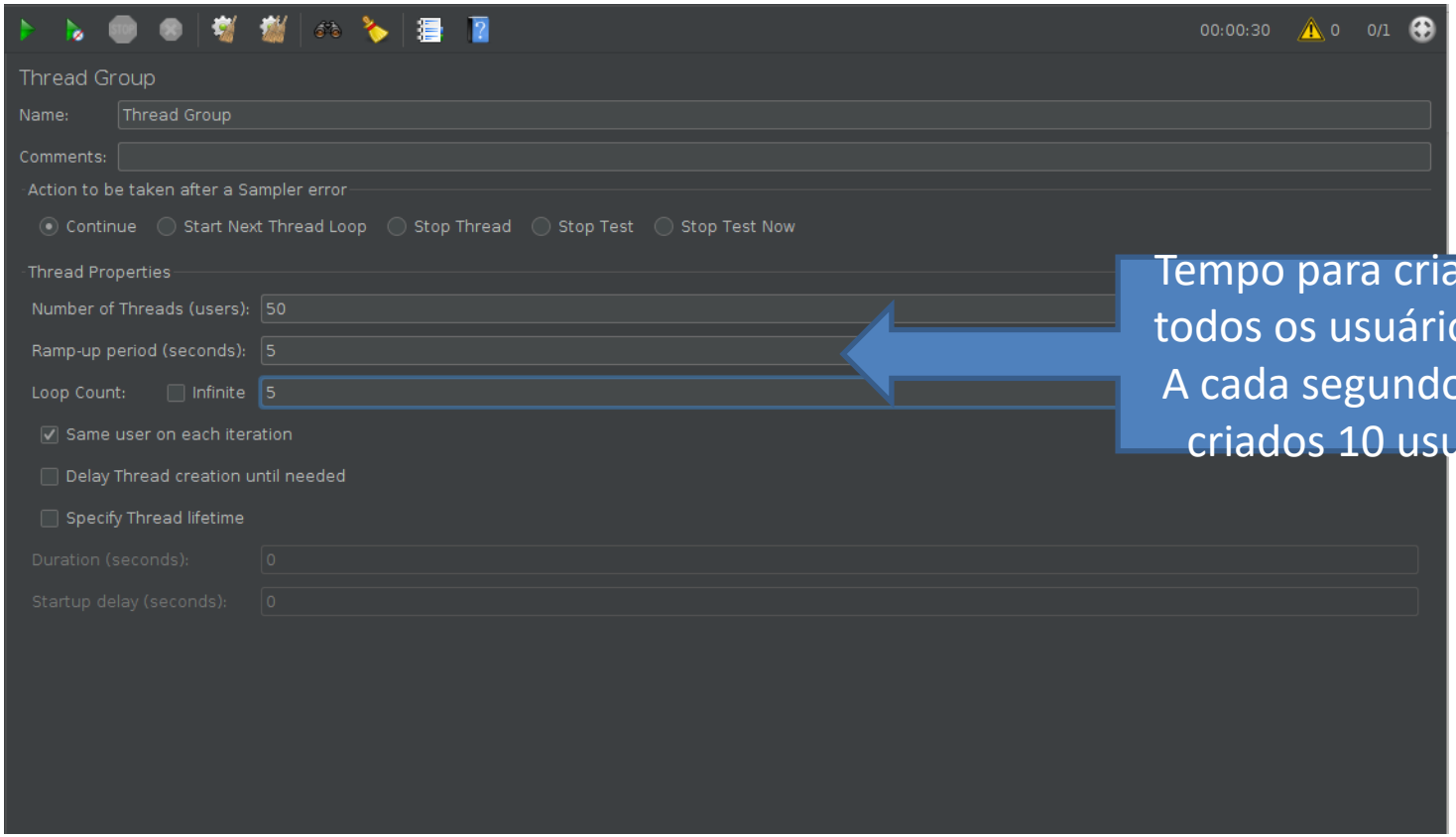


Grupo de Usuários



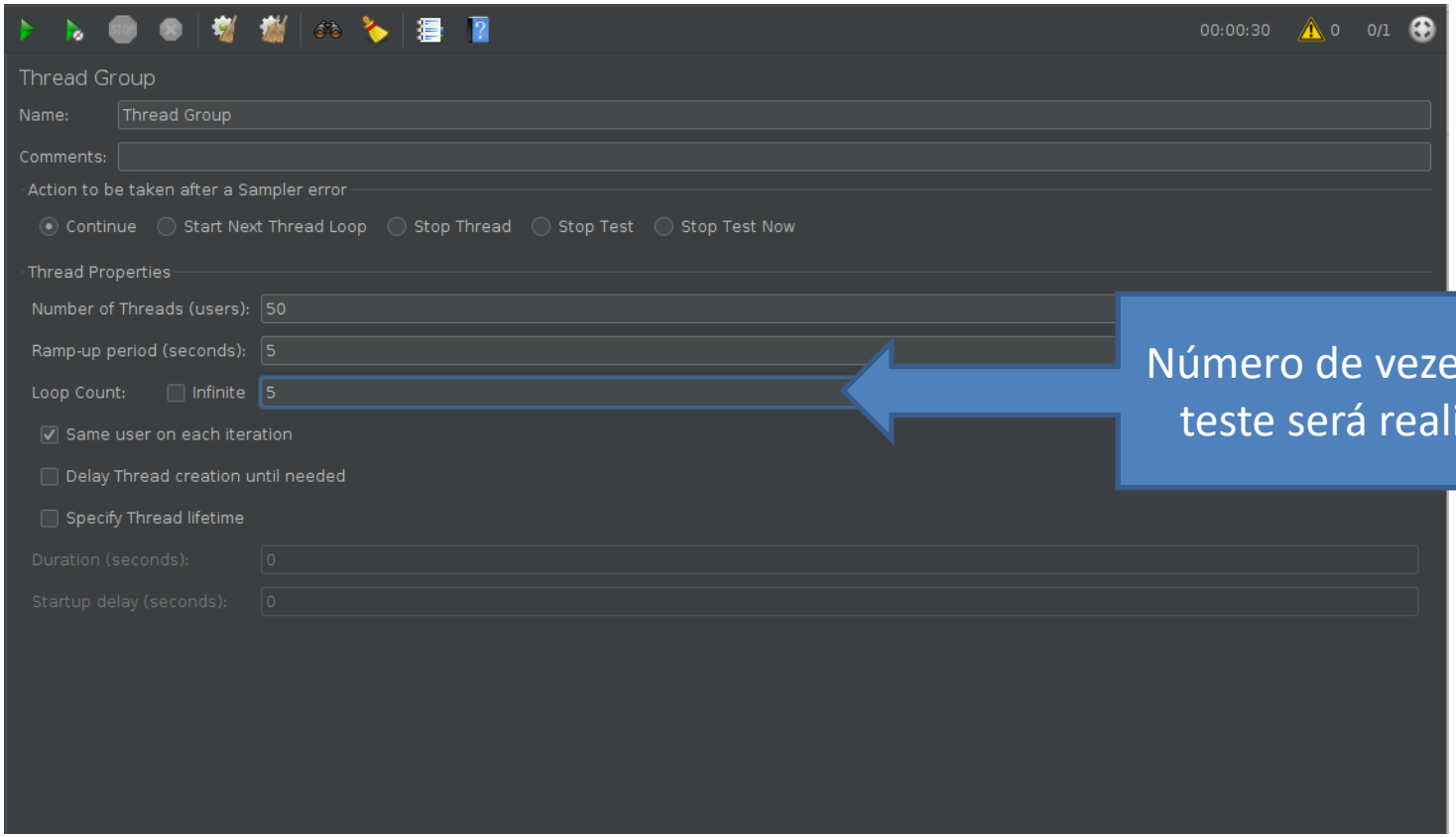
Total de Usuários serão criados

Grupo de Usuários



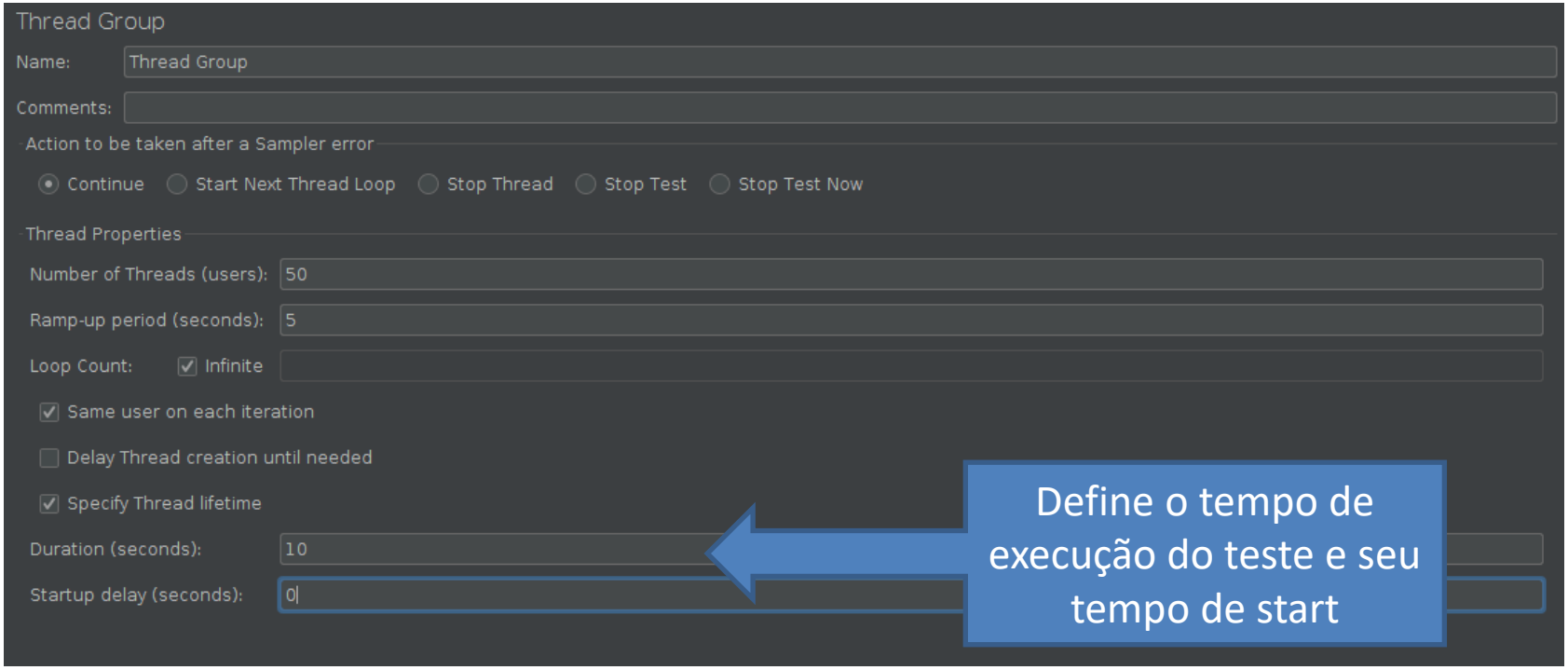
Tempo para criação de todos os usuários. Exe. A cada segundo serão criados 10 usuários

Grupo de Usuários



Número de vezes que o teste será realizado.

Grupo de Usuários



Parameters

Attribute	Description	Required
Name	Descriptive name for this element that is shown in the tree.	
Action to be taken after a Sampler error	Determines what happens if a sampler error occurs, either because the sample itself failed or an assertion failed. The possible choices are: • `Continue` - ignore the error and continue with the test • `Start Next Thread Loop` - ignore the error, start next loop and continue with the test • `Stop Thread` - current thread exits • `Stop Test` - the entire test is stopped at the end of any current samples. • `Stop Test Now` - the entire test is stopped abruptly. Any current samplers are interrupted if possible.	No
Number of Threads	Number of users to simulate.	Yes
Ramp-up Period	How long JMeter should take to get all the threads started. If there are 10 threads and a ramp-up time of 100 seconds, then each thread will begin 10 seconds after the previous thread started, for a total time of 100 seconds to get the test fully up to speed.	Yes
Loop Count	Number of times to perform the test case. Alternatively, "`forever`" can be selected causing the test to run until manually stopped.	Yes, unless forever is selected
Delay Thread creation until needed	If selected, threads are created only when the appropriate proportion of the ramp-up time has elapsed. This is most appropriate for tests with a ramp-up time that is significantly longer than the time to execute a single thread. I.e. where earlier threads finish before later ones start. If not selected, all threads are created when the test starts (they then pause for the appropriate proportion of the ramp-up time). This is the original default, and is appropriate for tests where threads are active throughout most of the test.	Yes
Scheduler	If selected, enables the scheduler	Yes
Duration (seconds)	If the scheduler checkbox is selected, one can choose a relative end time. JMeter will use this to calculate the End Time.	No
Startup delay (seconds)	If the scheduler checkbox is selected, one can choose a relative startup delay. JMeter will use this to calculate the Start Time.	No

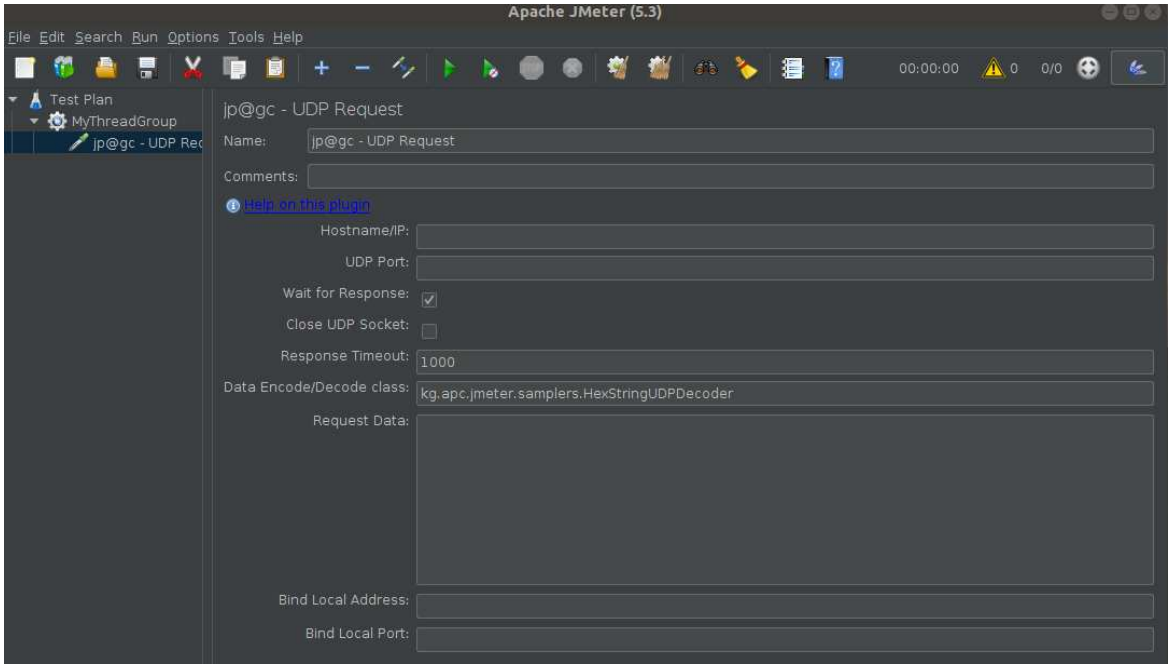
Requisições (Samplers)

Nele temos disponíveis várias opções de requisições, como FTP, SOAP/XML-RCP, Java, HTTP, entre outros.

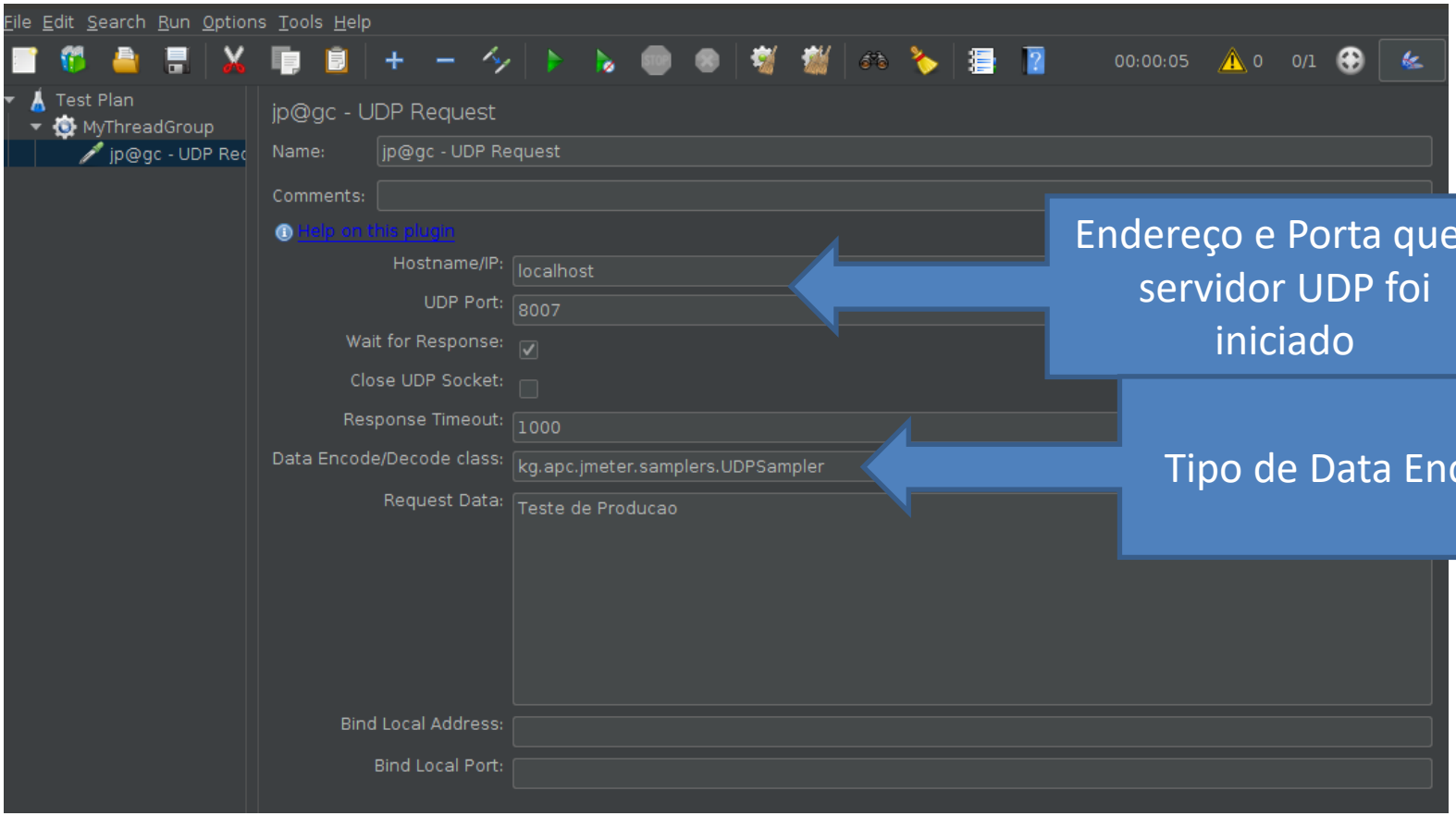
Temos várias opções de métodos para definir na requisição, porém os mais utilizados são GET e POST. O GET envia uma requisição ao servidor e ela é somente de leitura ou consulta. Já o POST também envia uma requisição ao servidor, porém ela pode enviar dados para inclusão ou edição de registros.

Criar Sampler UDP

- Para criar um Sampler UDP, clicar com botão direito no ThreadGroup, "add"-> "Sampler"-> "UDP Request".



Criar Sampler UDP



Criar Sampler UDP

Full Class Name	Comments
kg.apc.jmeter.samplers.HexStringUDPDecoder	This is most useful implementation, converts data from/to HEX-encoded sequences. For example, `6a6d6574657220706c7567696e73` corresponds to `jmeter plugins`.
kg.apc.jmeter.samplers.DNSJavaDecoder	Request data must contain three fields, separated with spaces: name, type, class. Example: `www.com. A IN`. Response data converted to text using DNSJava. Request flags can be set using +/- integer value on new line, eg `7` sets *reqursion desired* flag.
kg.apc.jmeter.samplers.UDPSampler	This implementation used as default when no valid class name specified in GUI. It makes no conversion on data.

Análise de resultados

A análise dos resultados é a parte mais importante do teste, afinal se passarmos resultados falsos para o cliente ou supervisores, o sistema apresentará erros em produção e teremos retrabalho para a equipe do projeto.

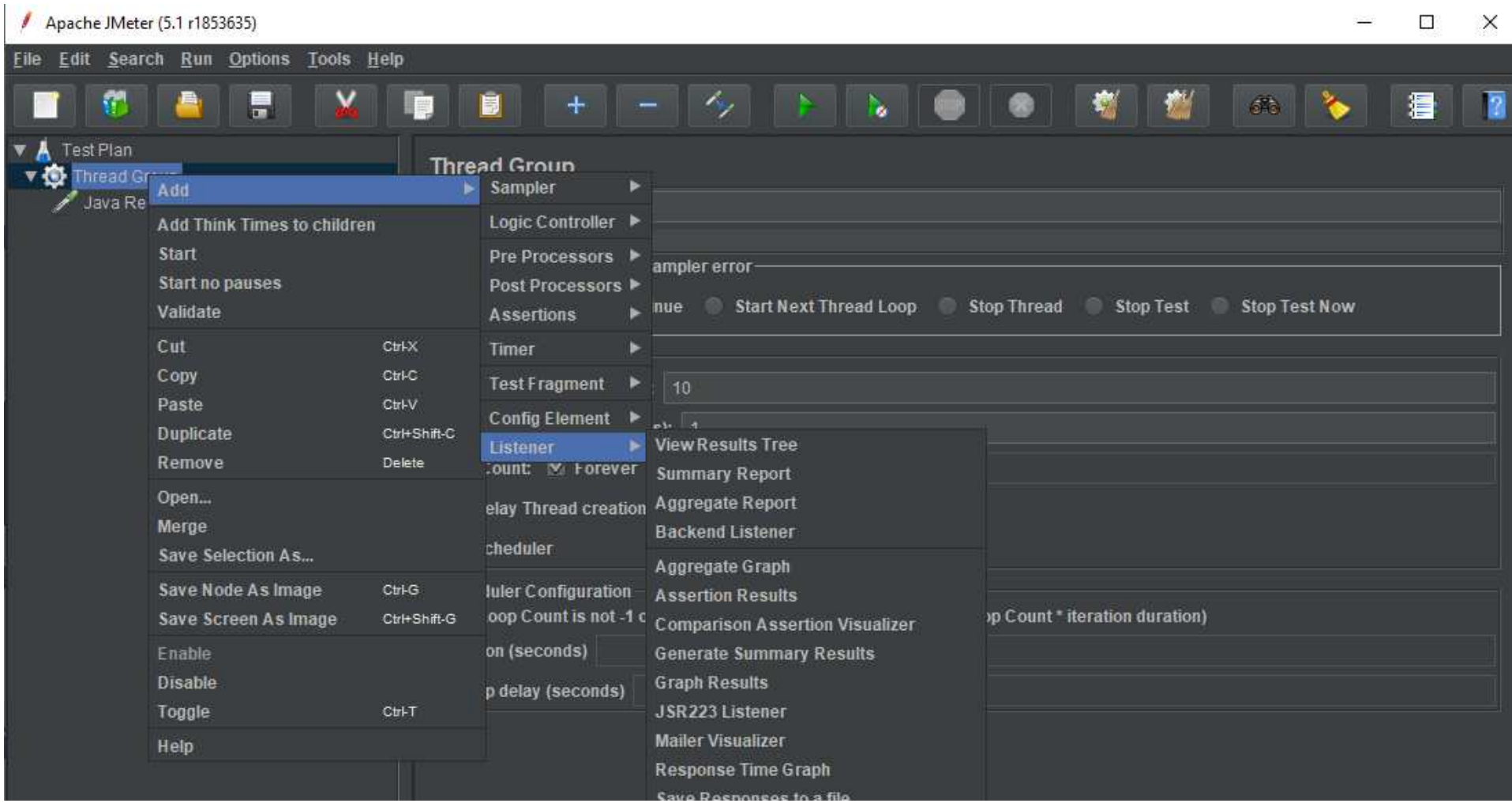
Para monitorar os testes, o JMeter disponibiliza vários componentes que são chamados de ouvintes(Listener) que podemos adicionar ao plano de teste, em cada grupo de usuário ou em cada requisição.

Ouvintes (Listeners)

Para visualizar o resultado da execução e das asserções, devemos utilizar o componente ouvinte. Com o resultado das asserções, podemos inserir condições para o JMeter realizar, por exemplo, quando uma asserção falhar para a execução do teste com aquele usuário.

Também é feito a captura das informações de cada requisição feita por cada usuário durante o teste.

Ouvintes

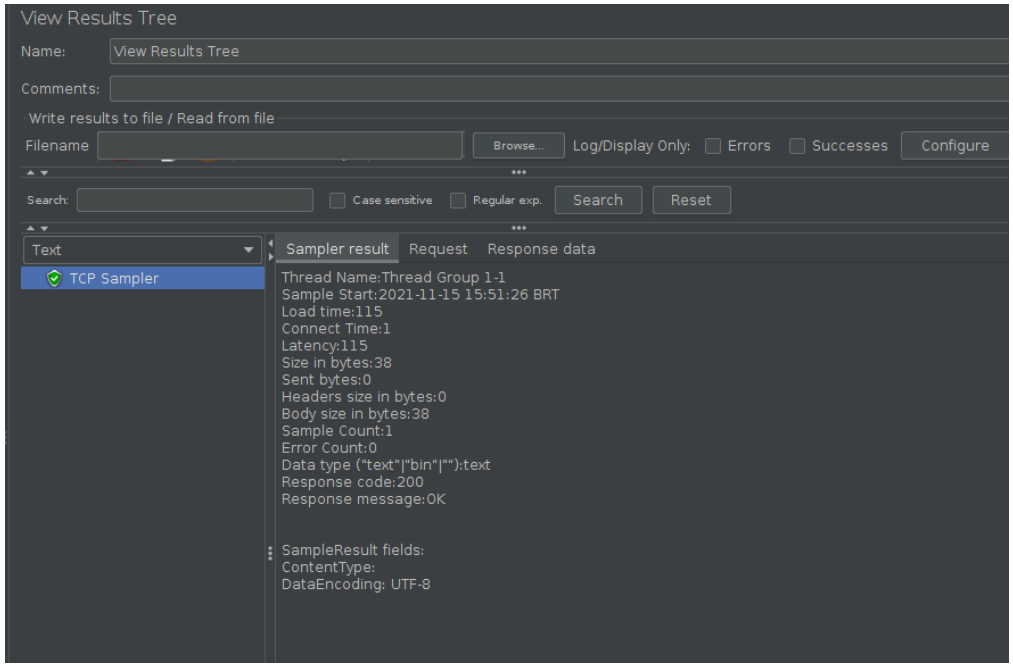

Apache JMeter (5.1 r1853635)
File Edit Search Run Options Tools Help
Test Plan
Thread Group
Java Re
Add
Add Think Times to children
Start
Start no pauses
Validate
Cut Ctrl-X
Copy Ctrl-C
Paste Ctrl-V
Duplicate Ctrl+Shift-C
Remove Delete
Open...
Merge
Save Selection As...
Save Node As Image Ctrl-G
Save Screen As Image Ctrl+Shift-G
Enable
Disable
Toggle Ctrl-T
Help
Sampler
Logic Controller
Pre Processors
Post Processors
Assertions
Timer
Test Fragment
Config Element
Listener
View Results Tree
Summary Report
Aggregate Report
Backend Listener
Aggregate Graph
Assertion Results
Comparison Assertion Visualizer
Generate Summary Results
Graph Results
JSR223 Listener
Mailer Visualizer
Response Time Graph
ampler error
nue
Start Next Thread Loop
Stop Thread
Stop Test
Stop Test Now
10
Forever
elay Thread creation
cheduler
uler Configuration
oop Count is not -1
on (seconds)
p delay (seconds)
p Count * iteration duration)

View Results Tree(Listeners)

- A árvore de resultados mostra uma lista de todas as respostas das amostras, permitindo que você visualize a resposta de qualquer amostra.
- Além de exibir a resposta, você também pode ver o tempo que levou para obter essa resposta e alguns códigos de resposta.

View Results Tree(Listeners)

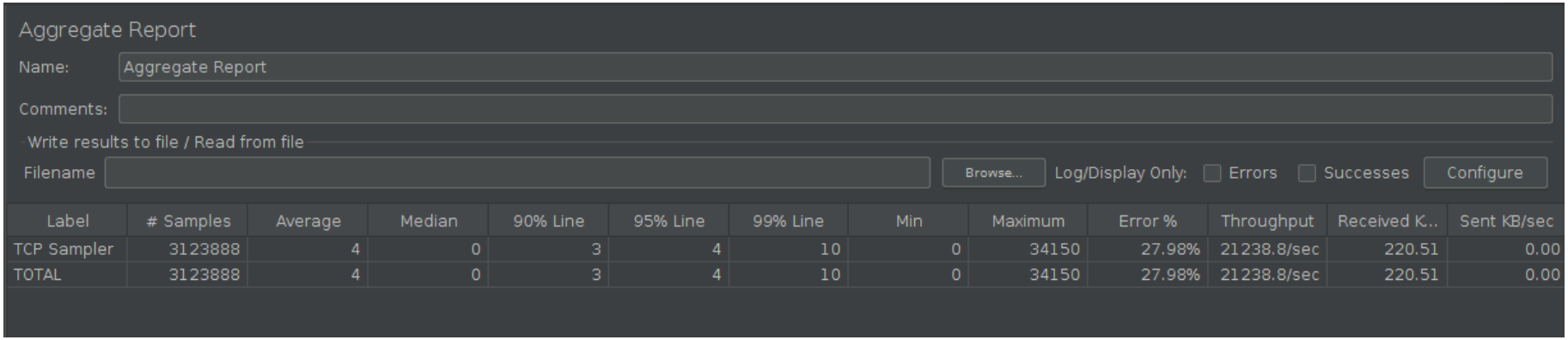
- Use para verificar se o fluxo de execução do seu teste está funcionando de acordo com o planejado



Aggregate Report (Listeners)

- O relatório agregado cria uma linha de tabela para cada solicitação com nome diferente em seu teste.
- Para cada solicitação, ele totaliza as informações de resposta e fornece a contagem de requisições, o tempo mínimo, máximo, médio, taxa de erro, taxa de transferência aproximada (requisições por segundo) e taxa de transferência em kilobytes por segundo.
- Quando o teste é concluído, a taxa de transferência corresponde à real durante toda a duração do teste.

Aggregate Report (Listeners)



Aggregate Report

Name: Aggregate Report

Comments:

Write results to file / Read from file

Filename Browse... Log/Display Only: Errors Successes Configure

Label	# Samples	Average	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received K...	Sent KB/sec
TCP Sampler	3123888	4	0	3	4	10	0	34150	27.98%	21238.8/sec	220.51	0.00
TOTAL	3123888	4	0	3	4	10	0	34150	27.98%	21238.8/sec	220.51	0.00

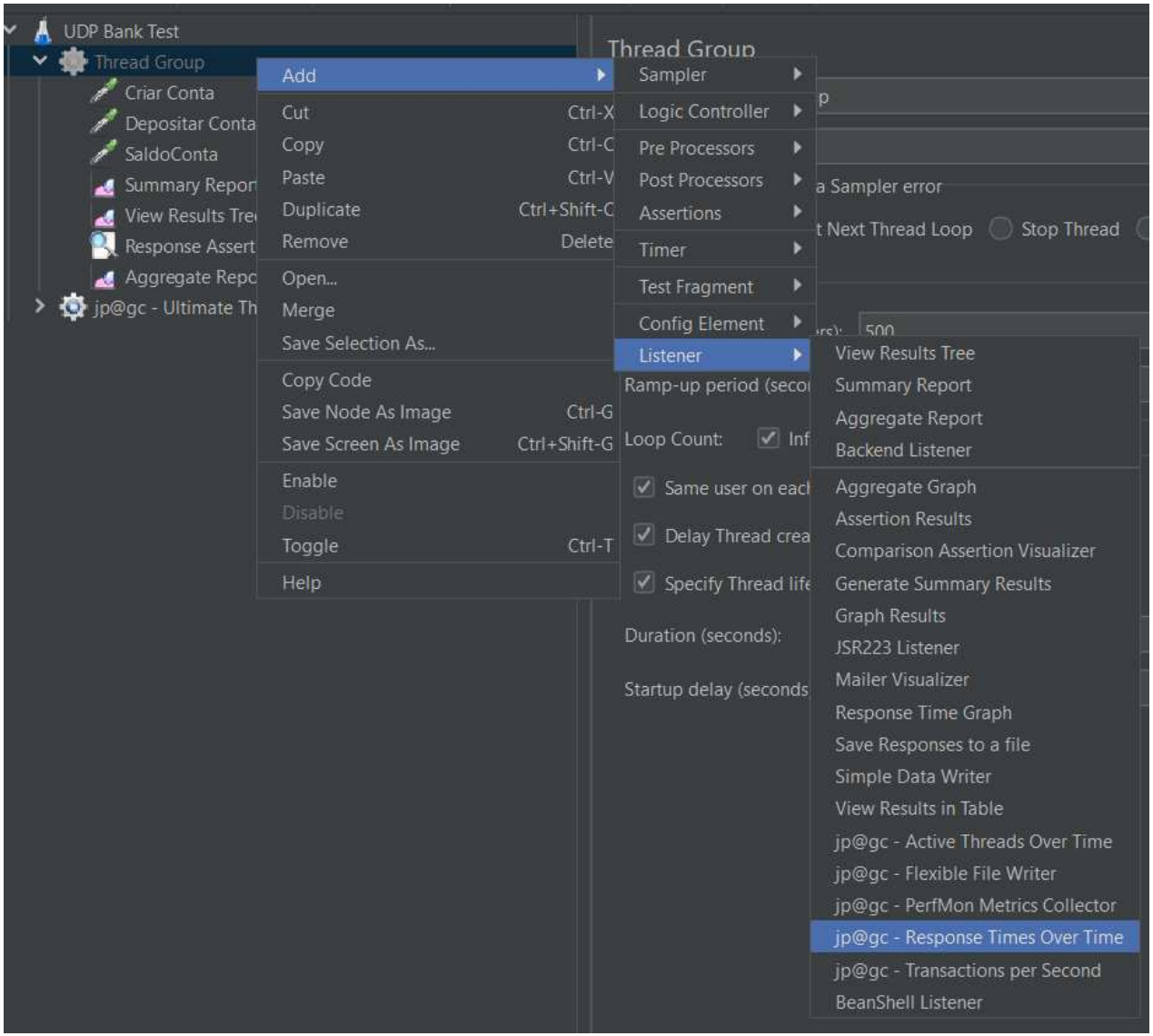
Aggregate Report (Listeners)

- **Samples** - Número de Requisições que chegou no servidor
- **Average** - Tempo médio de execução de cada requisição (em miliseconds)
- **Median** - Mediana, 50 % das requisições não levaram mais que este tempo(em miliseconds).
- **Min**- O menor tempo que uma requisição foi realizada.
- **Max**- O maior tempo que uma requisição foi realizada.
- **Error %** - Percentual de requisições com erro.

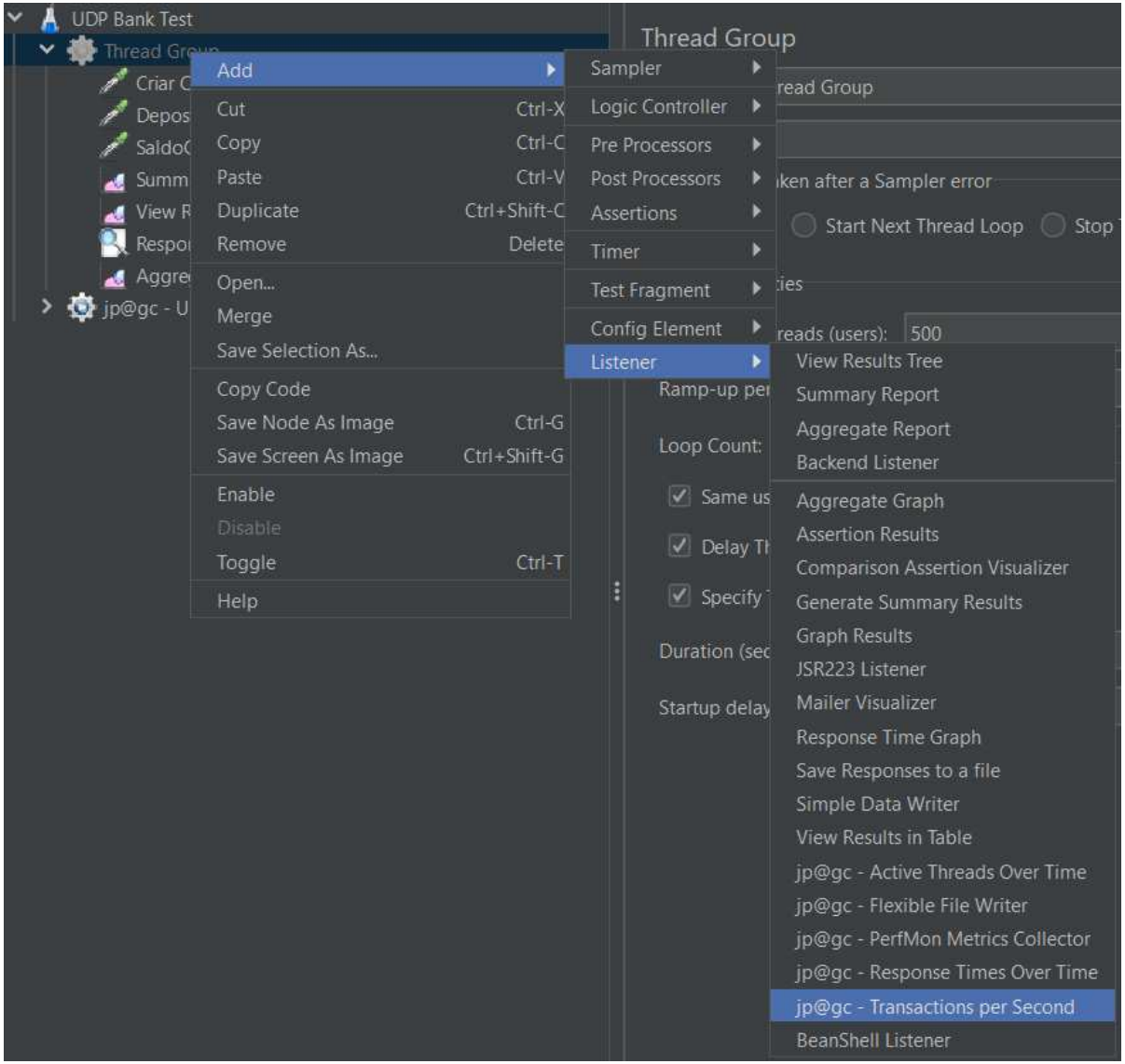
Aggregate Report (Listeners)

- **Throughput** - o Throughput é medido em requisições por second/minute/hour.
- **Received KB/sec** - o throughput é medido em termos de Kilobytes recebidos por segundo.
- **Sent KB/sec** - o throughput é medido em termos de Kilobytes enviados por segundo.

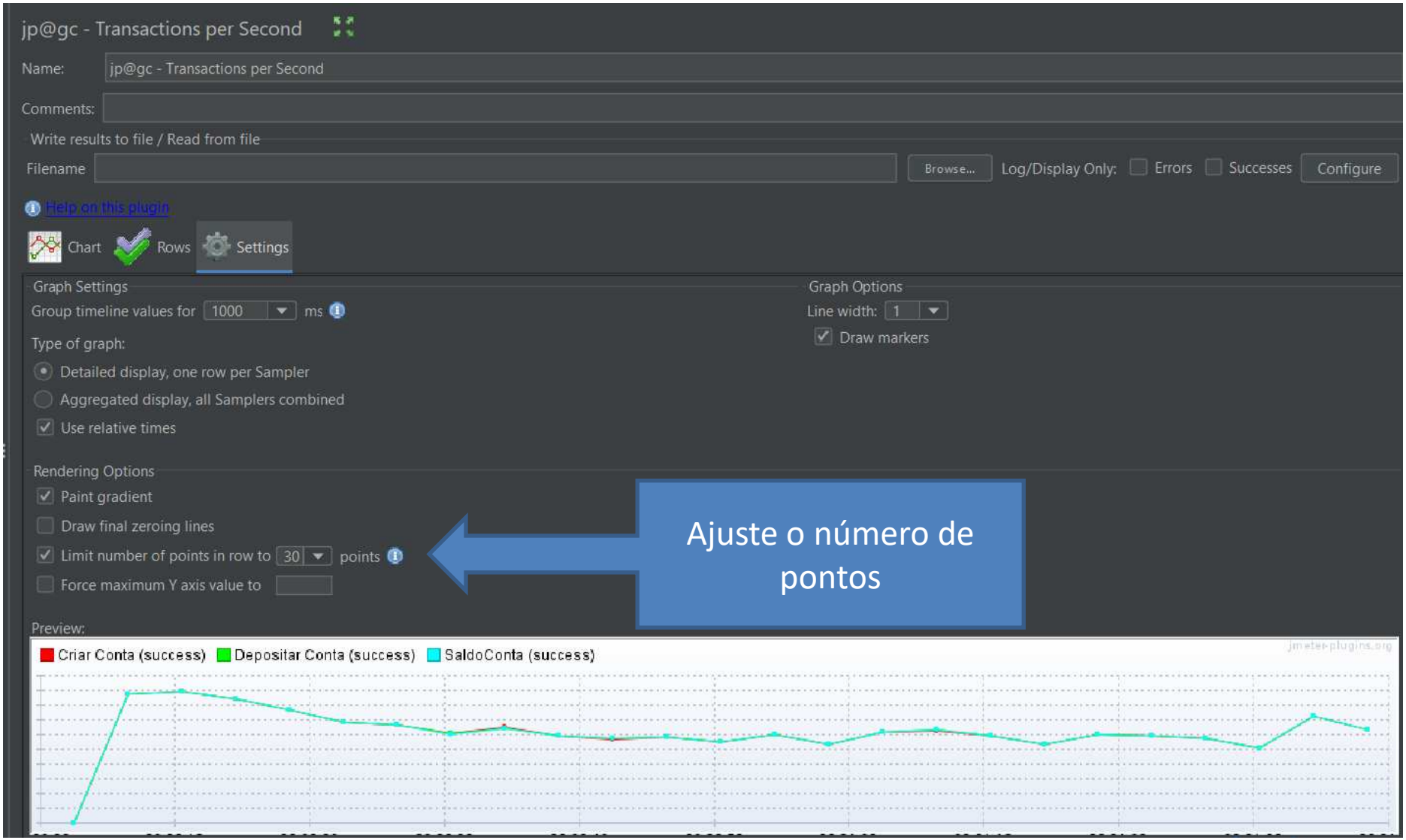
Response Times Over Time(Listeners)



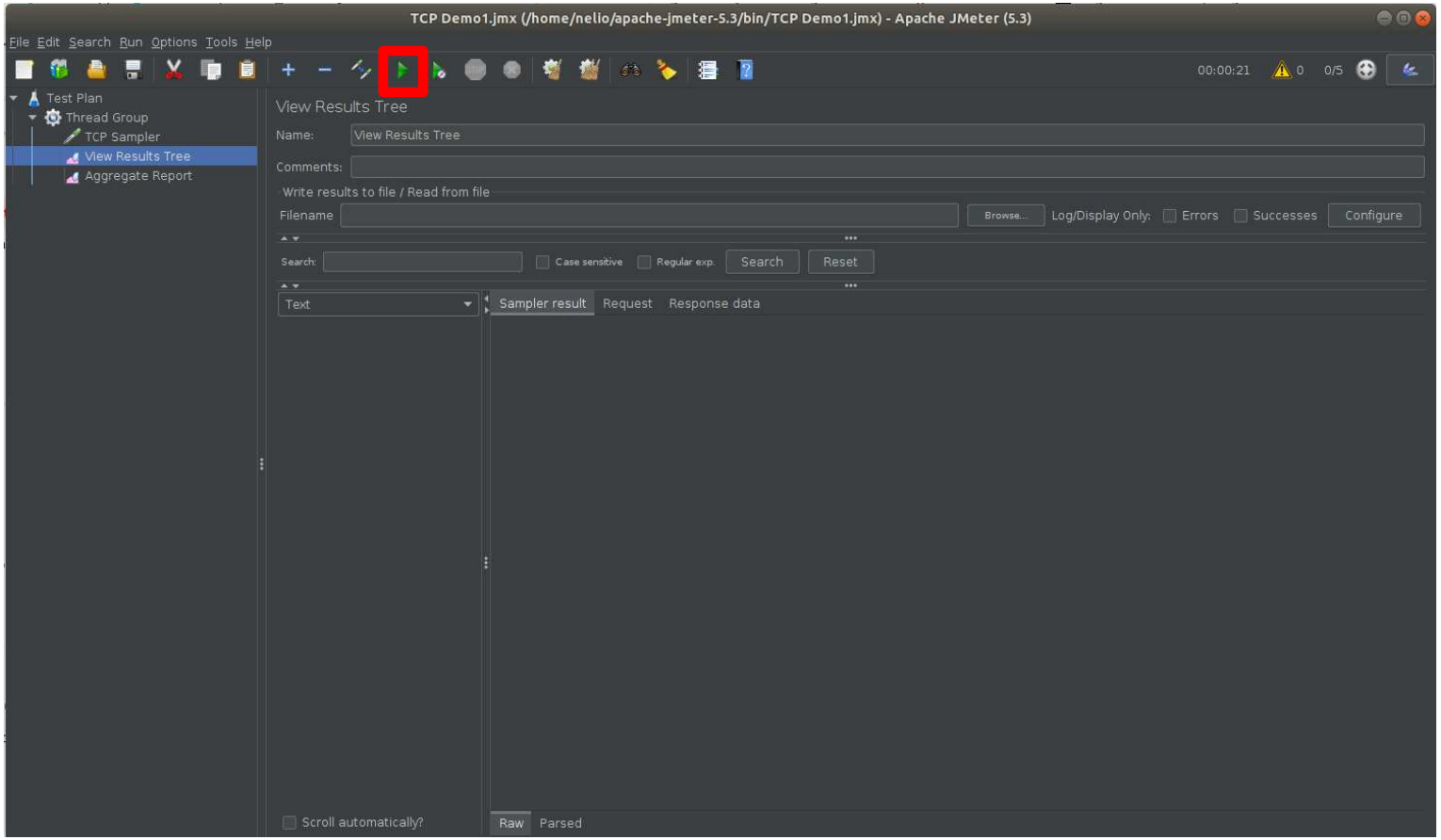
Transactions per Second(Listeners)



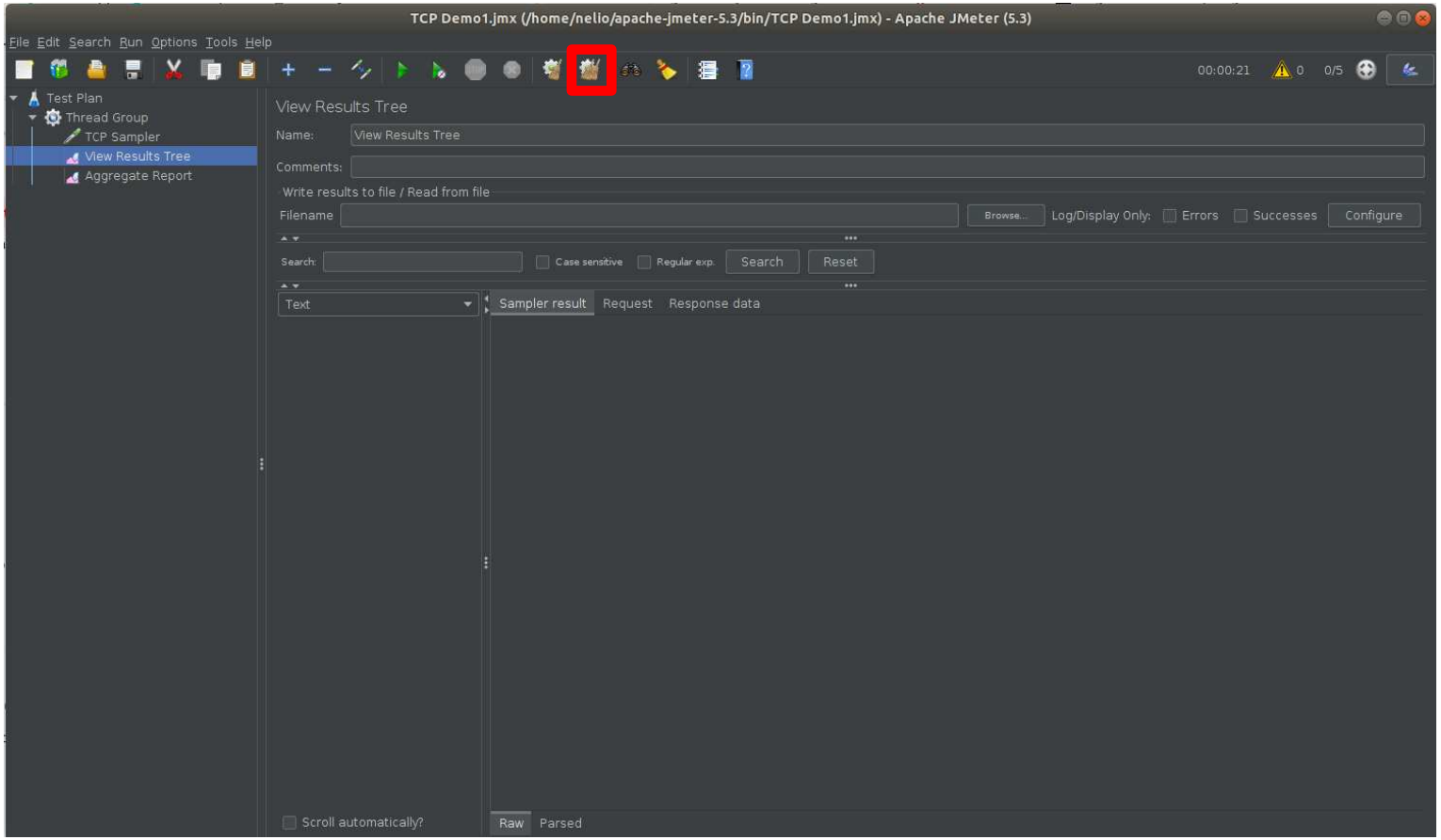
Transactions per Second(Listeners)



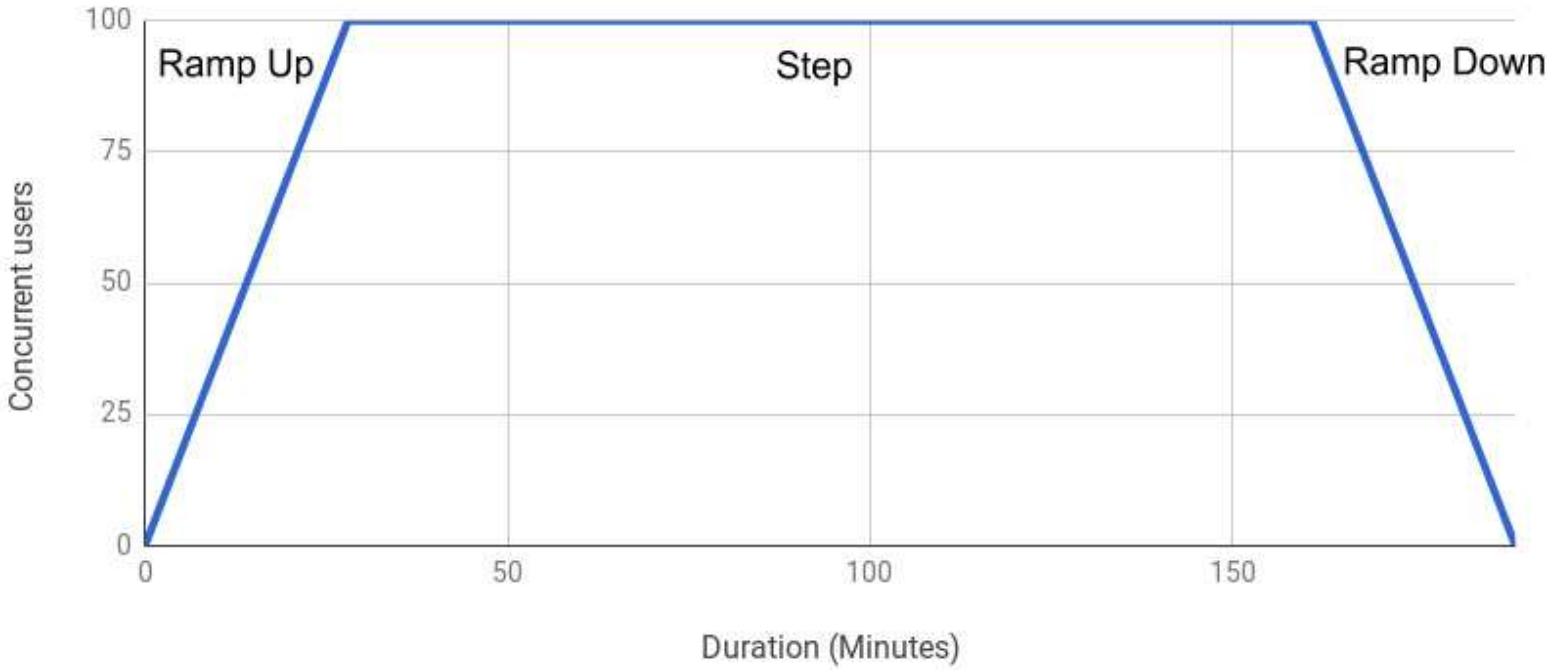
Execute o Teste



Limpar valores do Teste



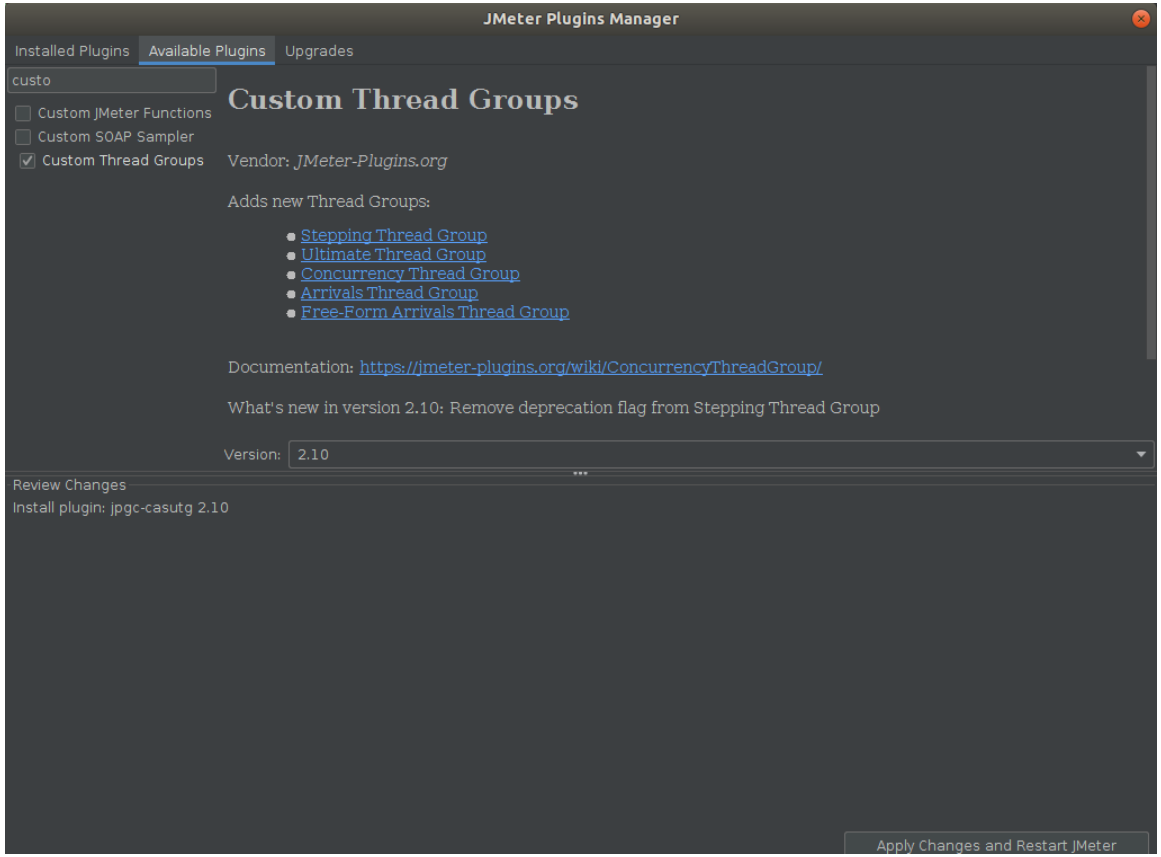
Estrutura de Um Caso de Teste



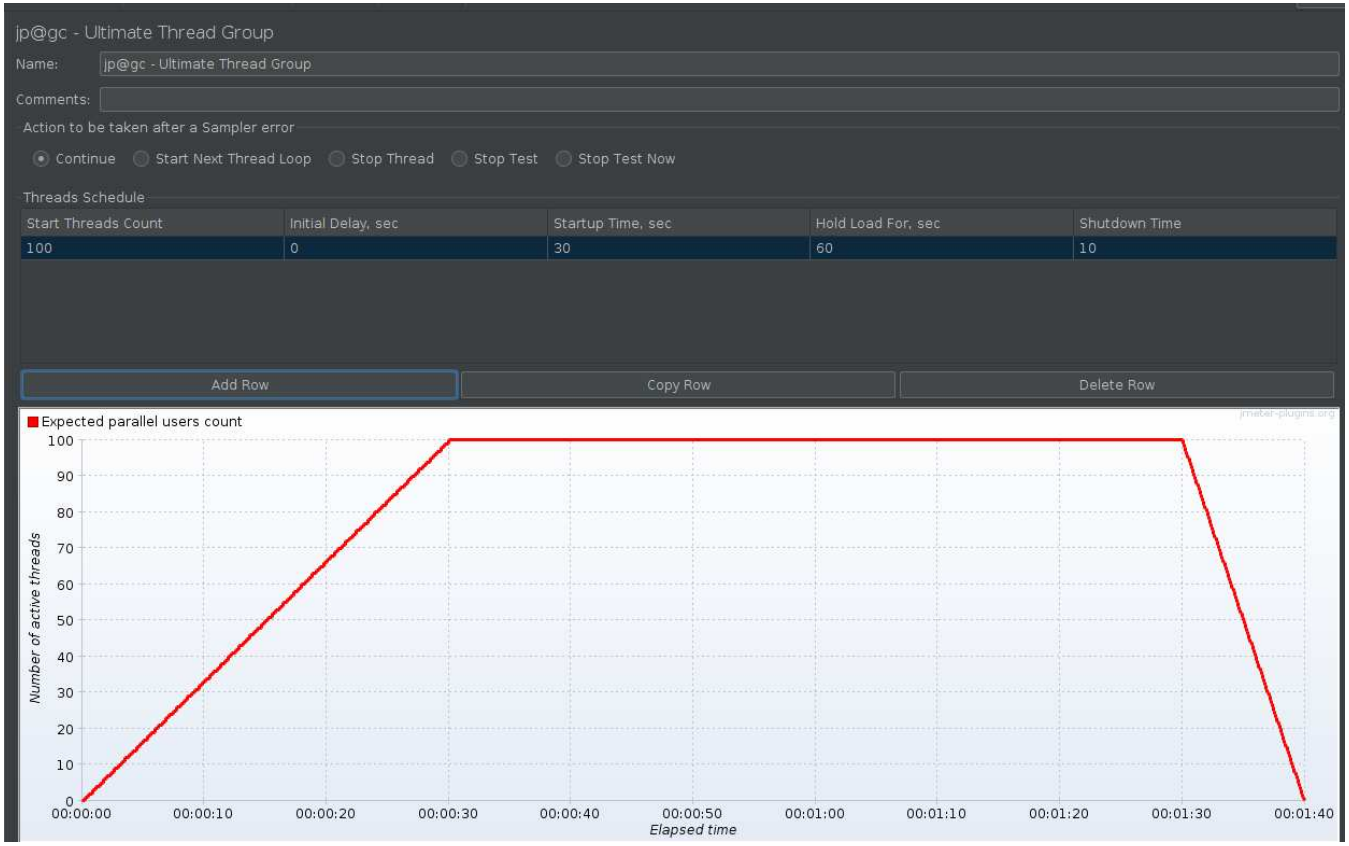
Estrutura de Um Caso de Teste

- **Ramp-Up:** Esse período corresponde à chegada gradual de usuários virtuais no sistema alvo. Um período de aquecimento permite que tanto o servidor alvo quanto os injetores aqueçam o compilador just in time (JIT) da máquina virtual Java (JVM) dos injetores.
- **Plateau/Step:** Após o ramp-up, chegamos a uma etapa. Sua duração deve ser longa o suficiente (pelo menos várias dezenas de minutos) para permitir uma análise relevante dos resultados.
- **Ramp-Down:** O ramp-down é o oposto do ramp-up. Esta é a parte do cenário onde o número de usuários diminui gradualmente.

Instalar um Novo Plugin para Threadgroup

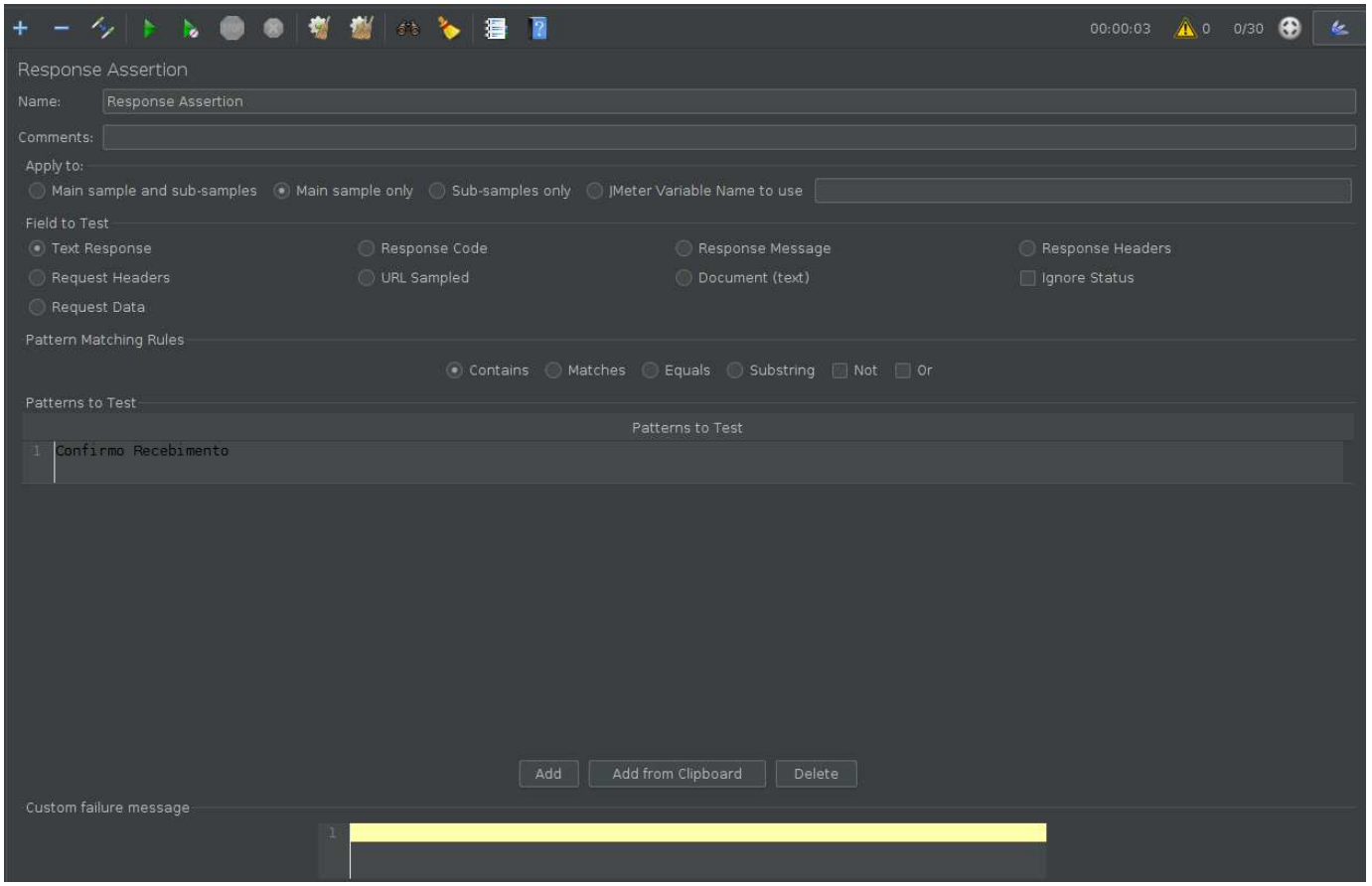


Instalar um Novo Plugin para Threadgroup

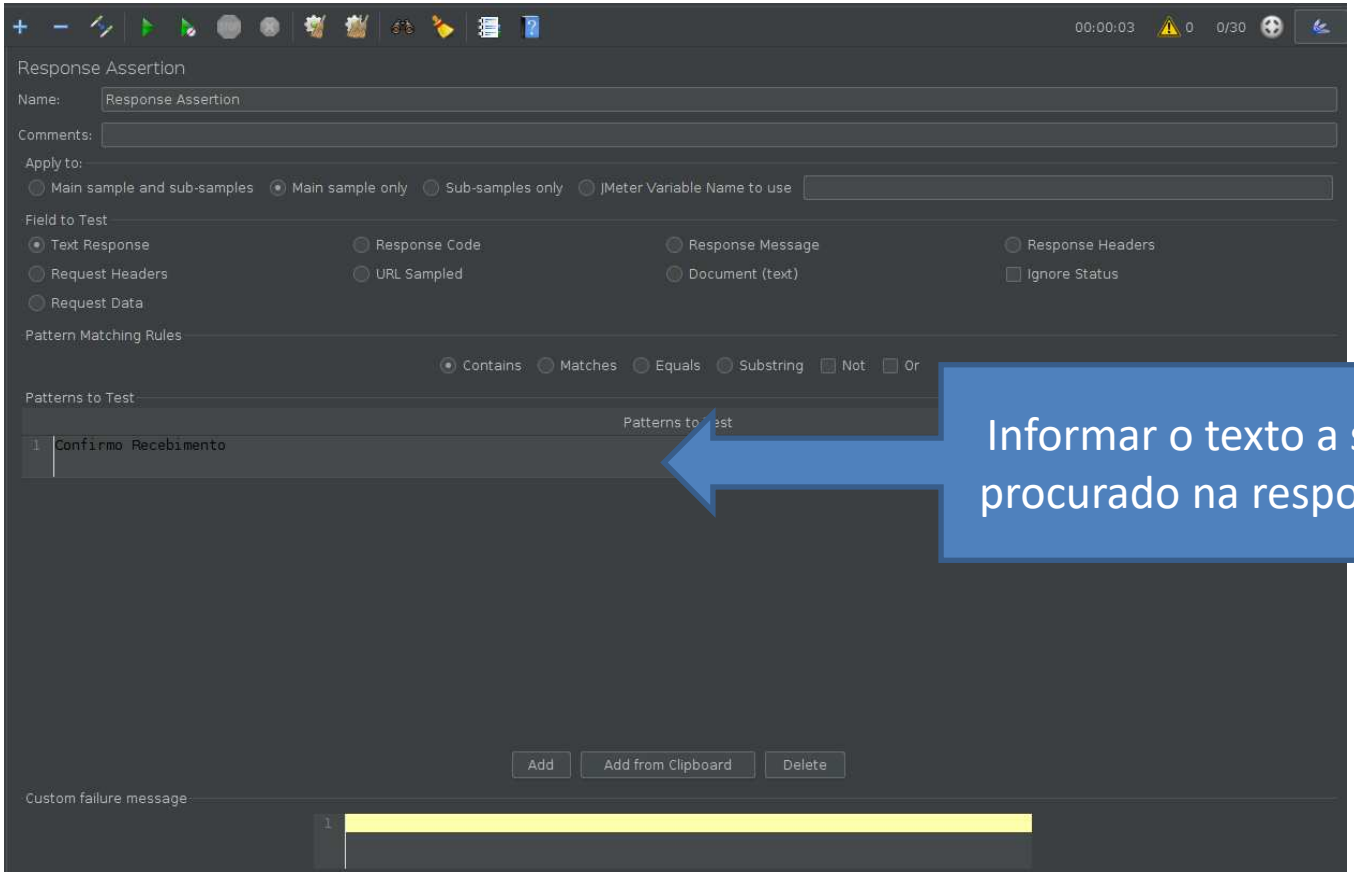


Incluindo Assertions

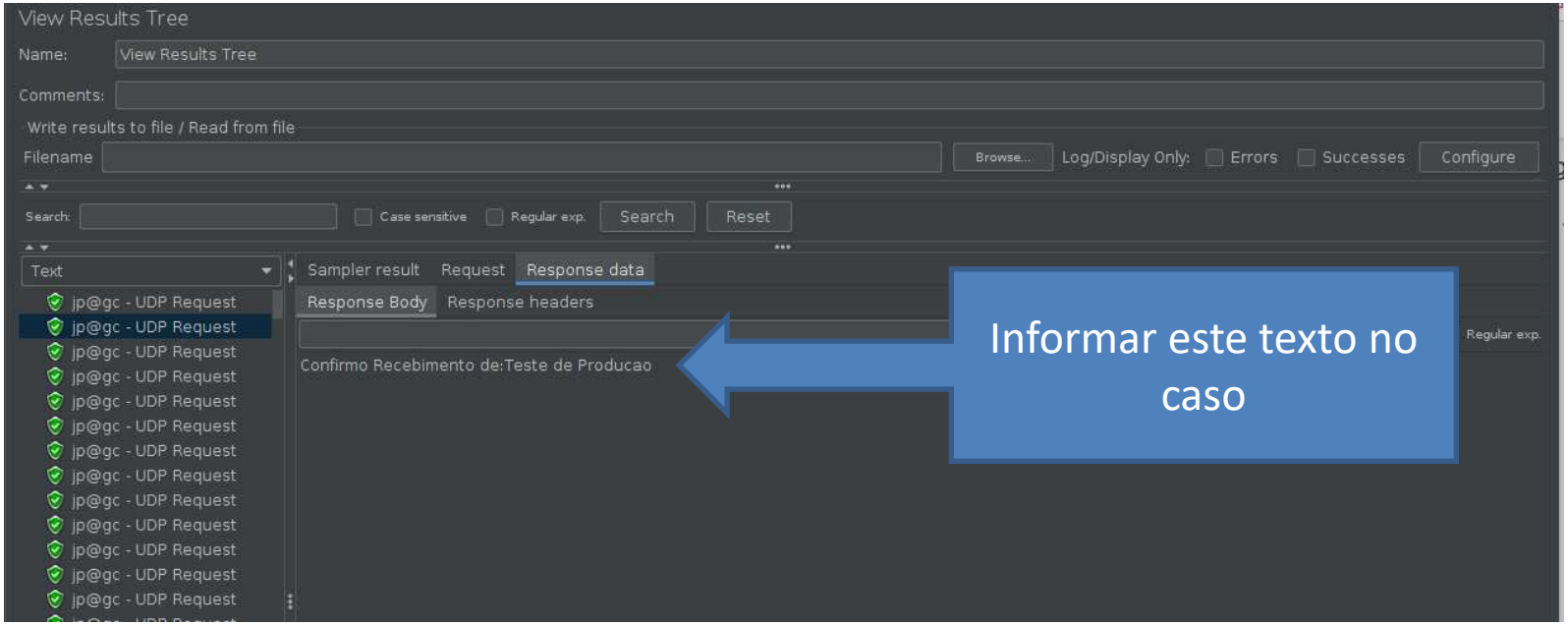
- Para criar um Assertion, clicar com botão direito no Sampler, "add"-> "Assertions"-> "Response Assertion".



Incluindo Assertions



Incluindo Assertions



Analisando os resultados

jp@gc - Ultimate Thread Group

Name: jp@gc - Ultimate Thread Group

Comments:

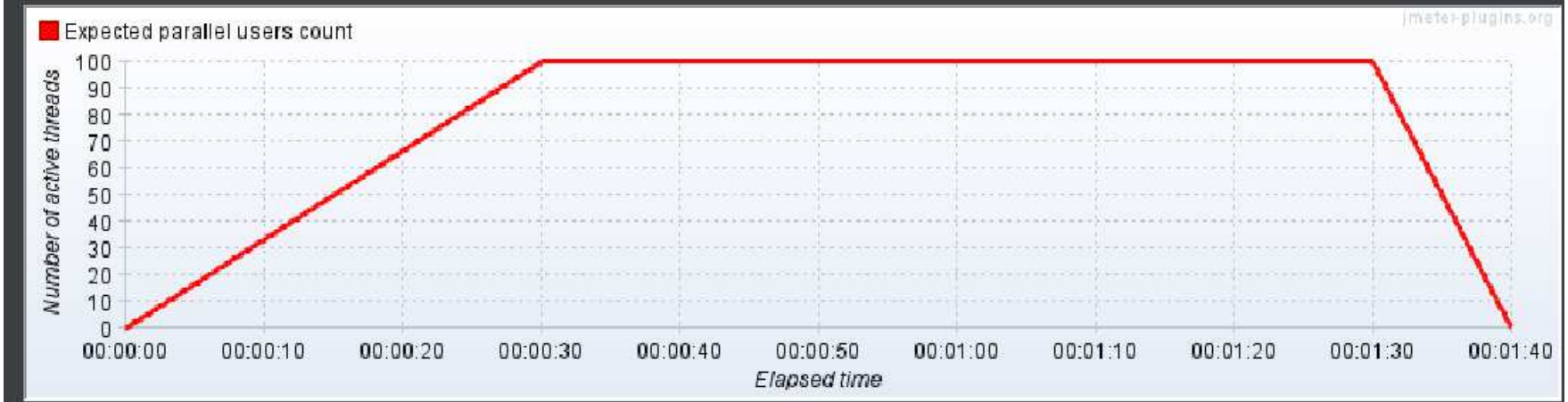
Action to be taken after a Sampler error

(•) Continue ( ) Start Next Thread Loop ( ) Stop Thread ( ) Stop Test ( ) Stop Test Now

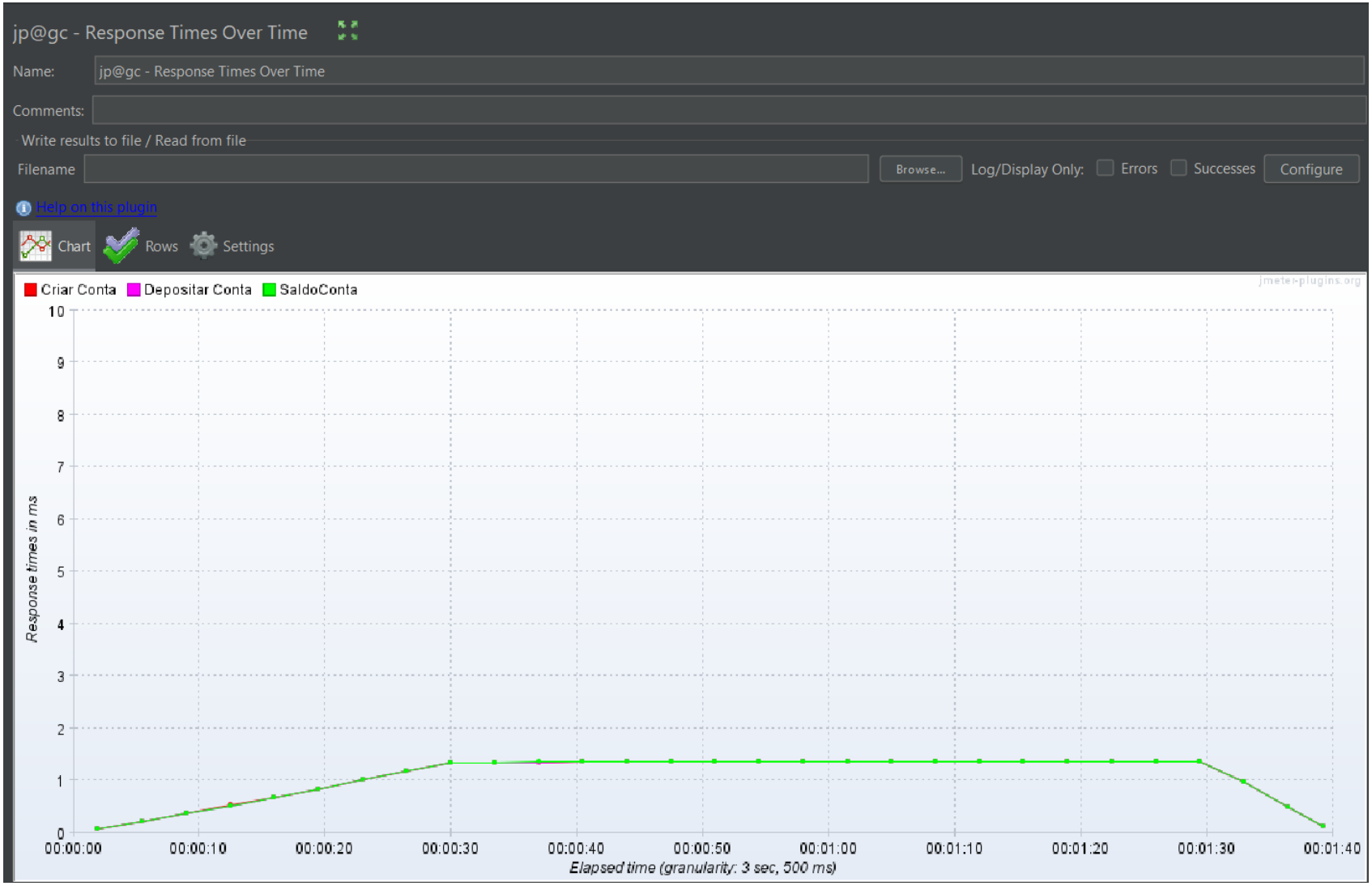
Threads Schedule

Start Threads Count	Initial Delay, sec	Startup Time, sec	Hold Load For, sec	Shutdown Time
100	0	30	60	10

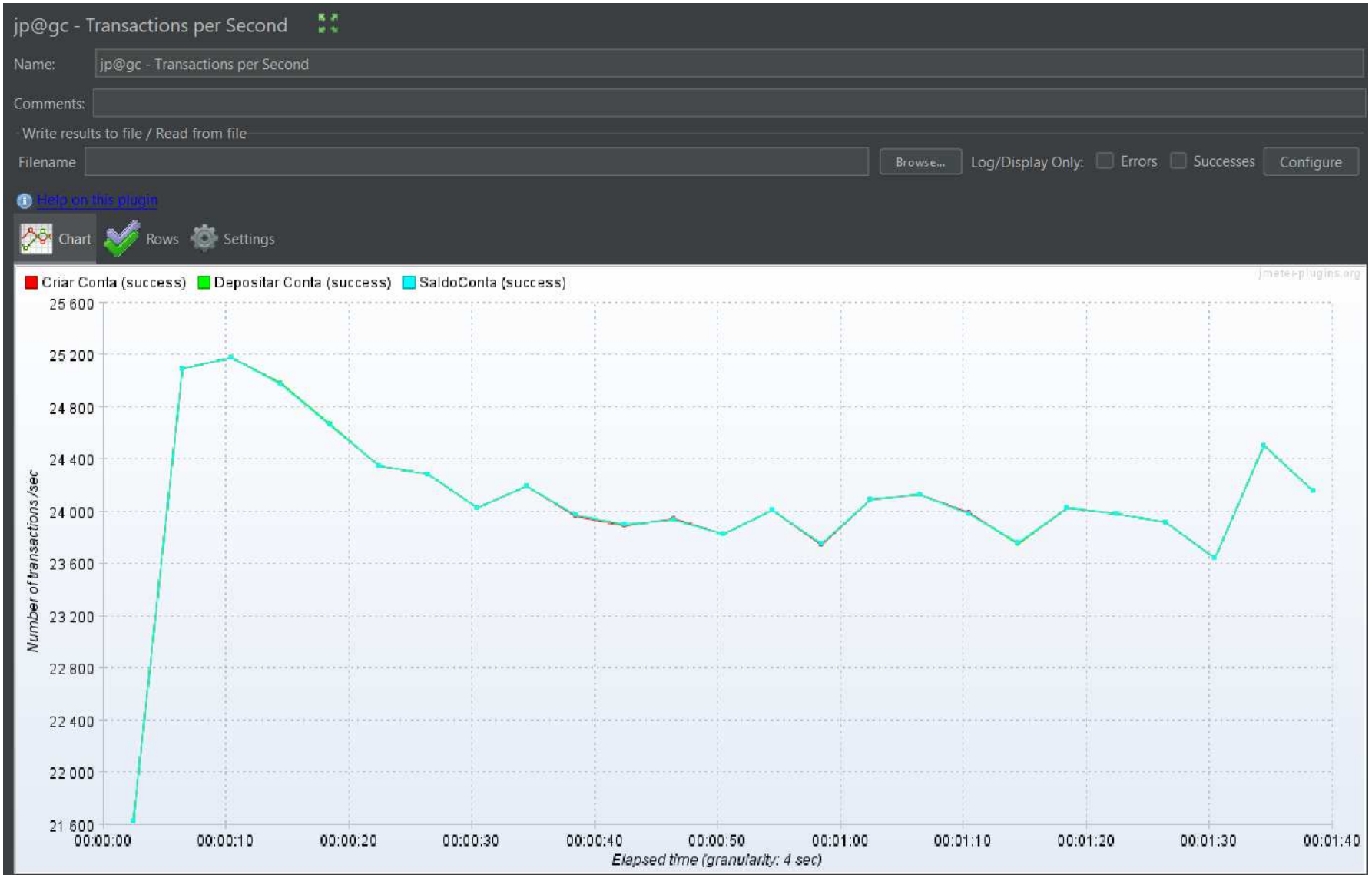
Add Row | Copy Row | Delete Row



Analisando os resultados



Analisando os resultados



Analisando os resultados

jp@gc - Ultimate Thread Group

Name: jp@gc - Ultimate Thread Group

Comments:

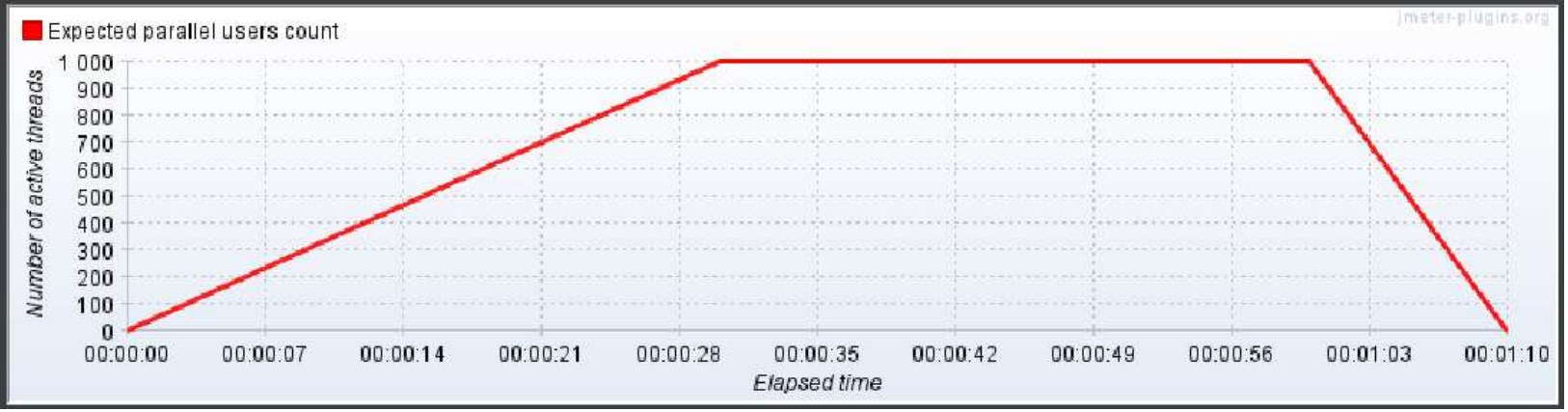
Action to be taken after a Sampler error

(•) Continue ( ) Start Next Thread Loop ( ) Stop Thread ( ) Stop Test ( ) Stop Test Now

Threads Schedule

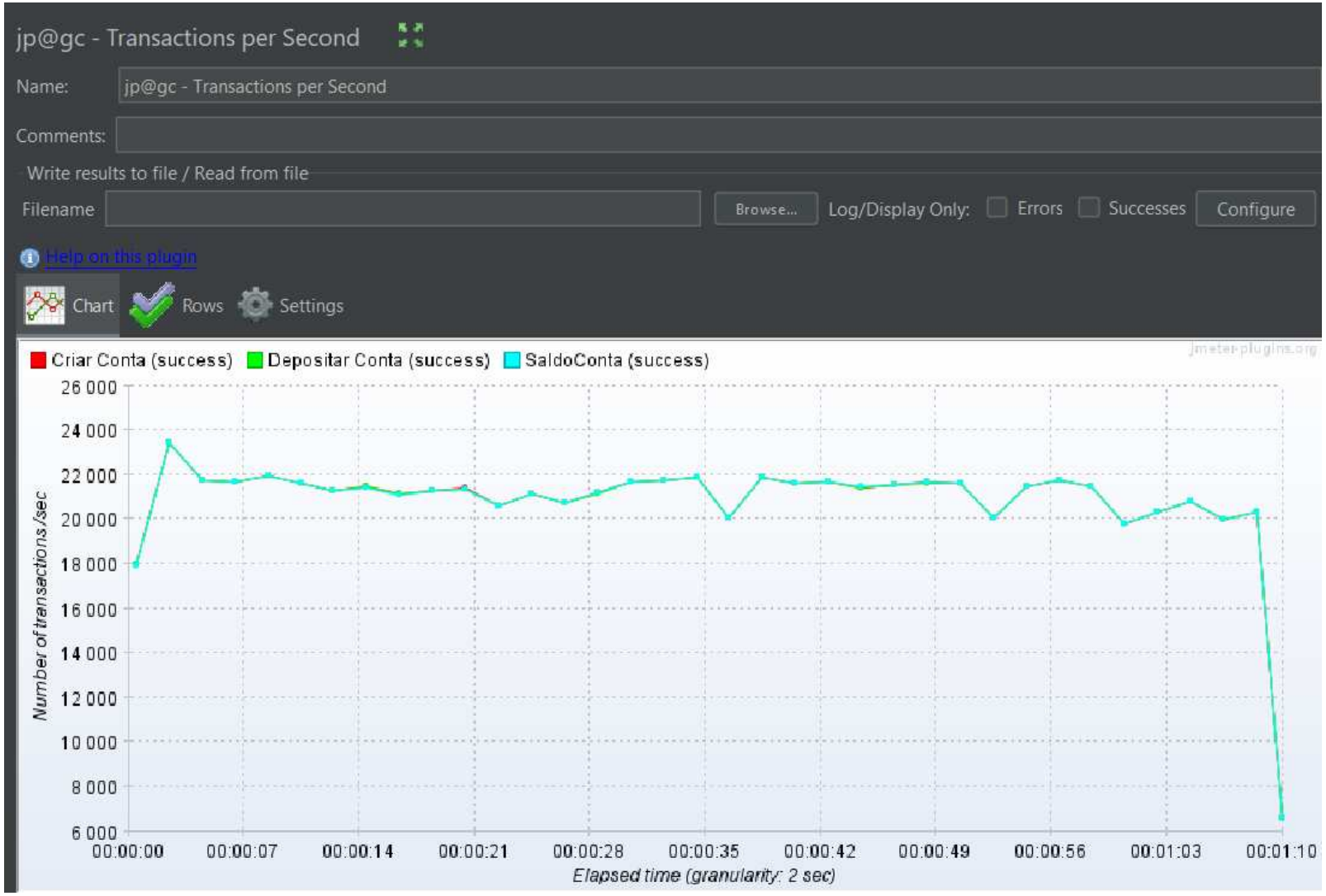
Start Threads Count	Initial Delay, sec	Startup Time, sec	Hold Load For, sec	Shutdown Time
1000	0	30	30	10

Add Row | Copy Row | Delete Row



Analisando os resultados

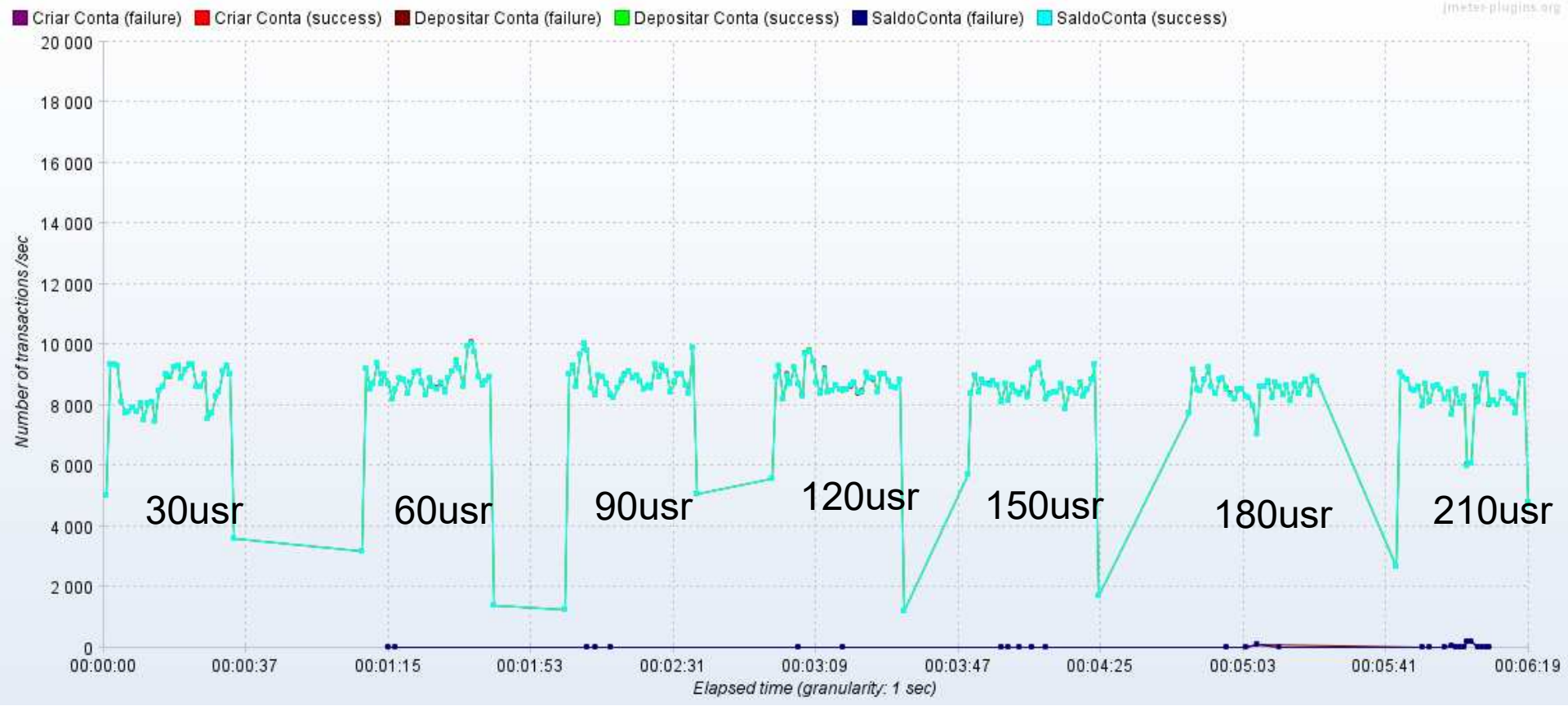
Analisando os resultados



Mono-thread


Criar Conta
Depositar Conta
SaldoConta
Response times in ms
20
18
16
14
12
10
8
6
4
2
0
00:00:00
00:00:37
00:01:15
00:01:53
00:02:31
00:03:09
00:03:47
00:04:25
00:05:03
00:05:41
00:06:19
Elapsed time (granularity: 500 ms)
30usr
60usr
90usr
120usr
150usr
180usr
210usr

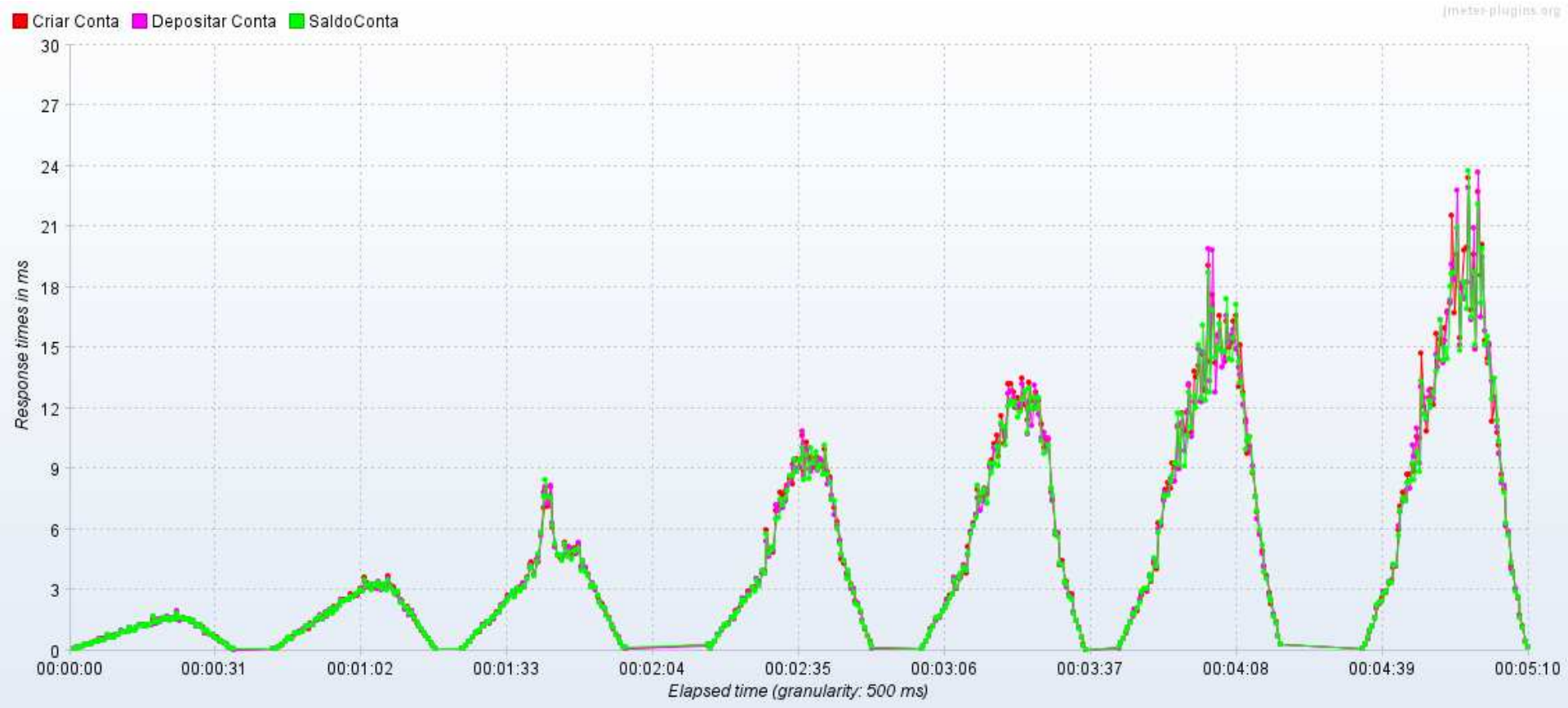
Mono-thread


Criar Conta (failure)
Criar Conta (success)
Depositar Conta (failure)
Depositar Conta (success)
SaldoConta (failure)
SaldoConta (success)
Number of transactions /sec
20 000
18 000
16 000
14 000
12 000
10 000
8 000
6 000
4 000
2 000
0
30usr
60usr
90usr
120usr
150usr
180usr
210usr
00:00:00
00:00:37
00:01:15
00:01:53
00:02:31
00:03:09
00:03:47
00:04:25
00:05:03
00:05:41
00:06:19
Elapsed time (granularity: 1 sec)

Referências

- https://www.devmedia.com.br/teste-de-performance-com-jmeter/34621
- https://newspaint.wordpress.com/2012/11/28/creating-a-java-sampler-for-jmeter/
- ERINLE, Bayo. **Performance testing with JMeter 2.9**. Packt Publishing Ltd, 2013.

Multi-thread



Multi-thread