

Relatório Simulador de Redes

Vynicius Pontes, Lucas Leão, Rodrigo Lima

Linguagem e IDE

O trabalho foi realizado com a linguagem Java na IDE Netbeans.

Execução

Para executar o projeto basta

- inserir os arquivos de texto com os nomes: physical-addresses com os endereços físicos, virtual-addresses com os endereços virtuais e as máscaras de rede, e o arquivo routing-rules com as linhas com regras de roteamento como descrito na especificação do projeto.
- executar o arquivo .jar com o comando "java -jar network-simulator.jar"
- Inserir o nome e porta da instância (Porta descrita no nome da pasta da instância)
- Inserir endereço IP para o qual se deseja conectar





Nome	Data de modificaç...	Tipo	Tamanho
 network-simulator	17/06/2017 02:23	Executable Jar File	43 KB
 physical-addresses	16/06/2017 23:40	Documento de Te...	1 KB
 routing-rules	17/06/2017 01:08	Documento de Te...	1 KB
 virtual-addresses	17/06/2017 00:07	Documento de Te...	1 KB

ilustração da disposição dos arquivos de cada instância

Arquitetura

A arquitetura do trabalho utiliza conceitos da engenharia de software e padrões GRASP para adotar responsabilidades a objetos.

O fluxo do programa inicia na classe **Program** que conduz a interação com o usuário e repassa para a classe **Simulator** que funciona como controlador do projeto. A classe **Simulator** possui instâncias de **Client** e **Server**, as quais ela repassa as requisições vindas da interação com o usuário.

A classe **Server** é responsável por iniciar a thread **MessageReceiver** que será a responsável por receber conexões, essa que ao receber uma conexão irá criar uma outra

thread chamada **MessageProcessor** que é responsável por imprimir os dados do datagrama recebido, checar qual o seu destino e caso a instancia do programa não seja o destino final ela repassa o datagrama para a classe **Router** para roteia o datagrama por alguma interface.

A classe **Client** é responsável por tratar as informações para o início de uma conexão e preparar o envio de uma mensagem inserida pelo usuário a uma determinada conexão. Já o envio físico de datagramas se dá pela classe **MessageDispatcher** que além disso realiza o particionamento dos datagramas em outros datagramas se necessário.

A classe **NodeCatalog** manipula e encapsula o conjunto de interfaces da instância do programa representadas pela classe **Node**. A classe **NodeCatalog** ainda recupera as informações das interfaces nos arquivos: physical-addresses e virtual-addresses.

A classe **Router** manipula uma lista regras de roteamento (representadas pela classe **RoutingRule**) e realiza o roteamento em implementando um algoritmo que consulta a lista de regras e decide a qual interface o datagrama será roteado. Após a decisão a classe **Router** envia o datagrama para que a classe **MessageDispatcher** inicie a conexão e faça o envio.

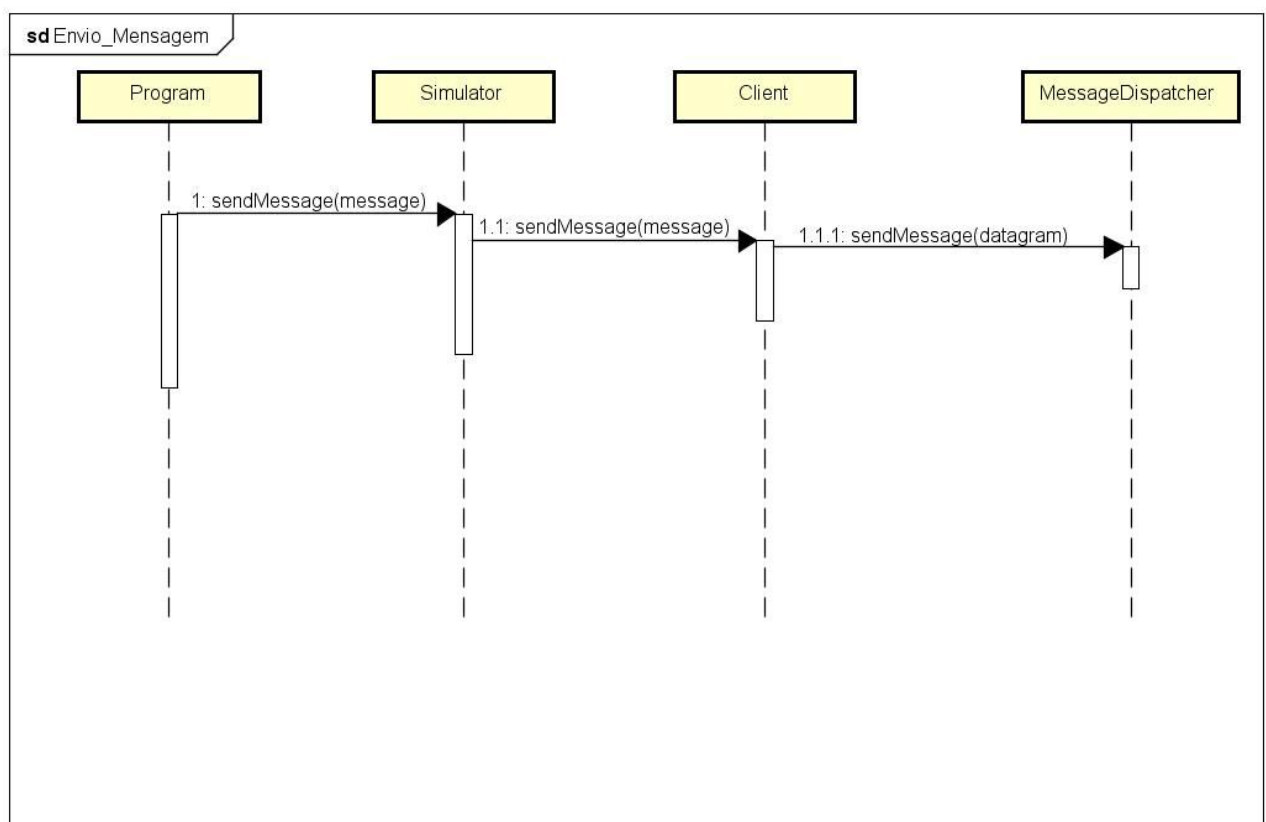
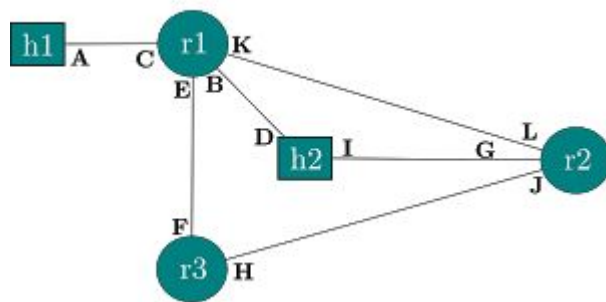


Diagrama de sequência do envio de uma mensagem



Endereços das Interfaces

A 10.0.0.2/24
 B 10.1.0.1/27
 C 10.0.0.1/24
 D 10.1.0.5/27
 E 10.4.0.2/30
 F 10.4.0.1/30
 G 10.2.0.1/30
 H 10.2.0.5/30
 I 10.2.0.2/30
 J 10.2.0.6/30
 K 10.3.0.1/30
 L 10.3.0.2/30

Topologia Proposta

```

C:\WINDOWS\system32\cmd.exe
C:\Users\Vynicius\Coding\Git\network-simulator\Apresenta
1 - 8050>java -jar network-simulator.jar
Digite o nome do server
R1
Digite o número de porta do server
8050
Server R1 executando na porta 8050
Digite um endereço virtual para iniciar uma conexão
  
```

```

C:\WINDOWS\system32\cmd.exe
C:\Users\Vynicius\Coding\Git\network-simulator\Apresenta
ion\R1 - 8051>java -jar network-simulator.jar
Digite o nome do server
R1
Digite o número de porta do server
8051
Server R1 executando na porta 8051
Digite um endereço virtual para iniciar uma conexão
  
```

```

C:\WINDOWS\system32\cmd.exe
C:\Users\Vynicius\Coding\Git\network-simulator\Apresenta
tion\R2 - 8053>java -jar network-simulator.jar
Digite o nome do server
R2
Digite o número de porta do server
8053
Server R2 executando na porta 8053
Digite um endereço virtual para iniciar uma conexão
  
```

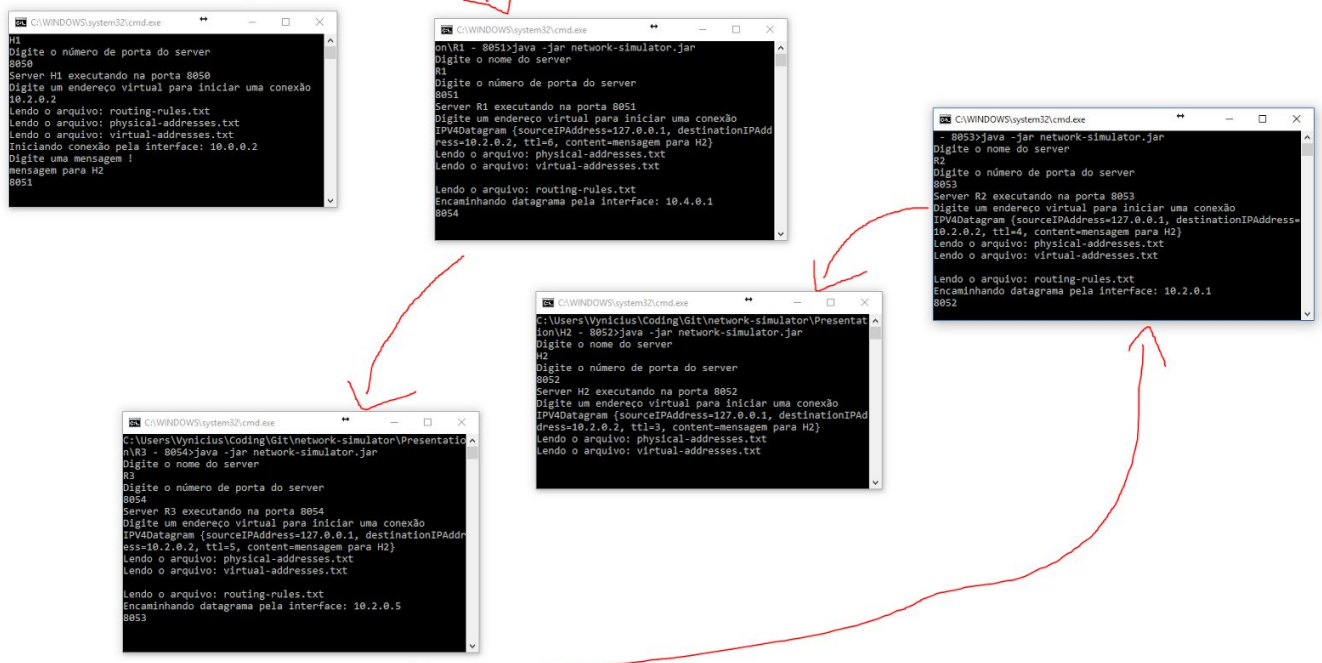
```

C:\WINDOWS\system32\cmd.exe
C:\Users\Vynicius\Coding\Git\network-simulator\Apresen
tation\R3 - 8054>java -jar network-simulator.jar
Digite o nome do server
R3
Digite o número de porta do server
8054
Server R3 executando na porta 8054
Digite um endereço virtual para iniciar uma conexão
  
```

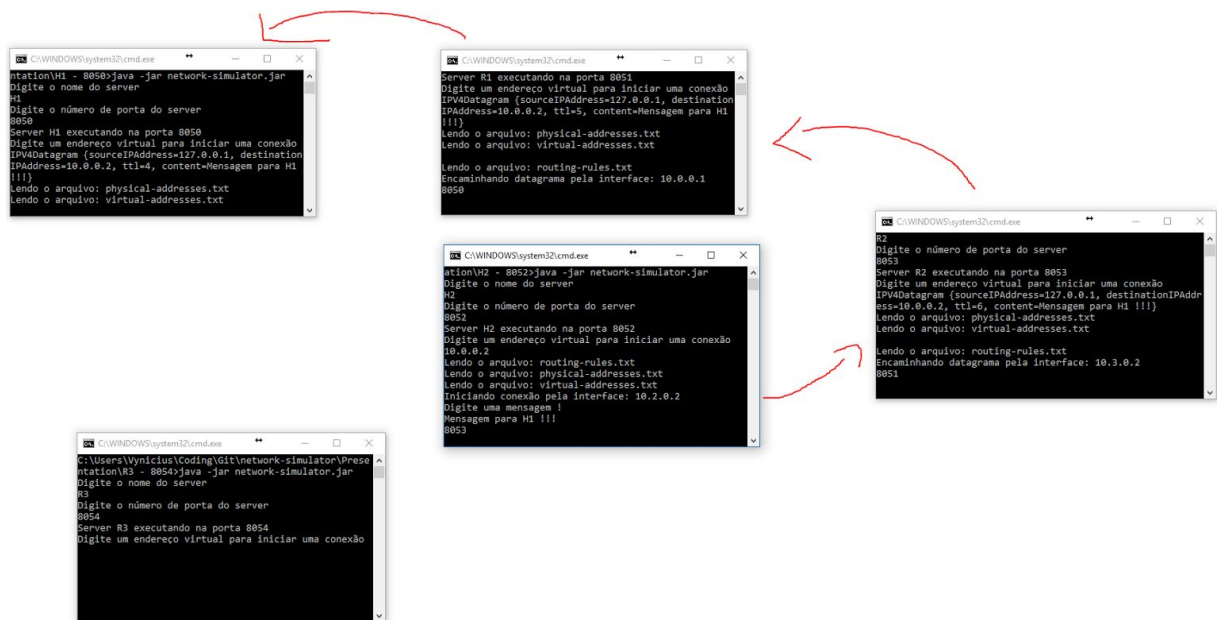
```

C:\WINDOWS\system32\cmd.exe
C:\Users\Vynicius\Coding\Git\network-simulator\Apresenta
tion\R2 - 8052>java -jar network-simulator.jar
Digite o nome do server
R2
Digite o número de porta do server
8052
Server R2 executando na porta 8052
Digite um endereço virtual para iniciar uma conexão
  
```

Setup Inicial



Mensagem de H1 para H2



Mensagem de H2 para H1