

**CIÊNCIA DA COMPUTAÇÃO**

ANDERSON DE LIMA G. DA SILVA - C98589-9

RICHARD WILLIAN R DIVINO - D074873

RODRIGO AFFONSO DE OLIVEIRA D074121

VITOR DOS SANTOS SILVA - C7704F8

**“DESENVOLVIMENTO DE UMA FERRAMENTA PARA COMUNICAÇÃO EM REDE”**

**ARARAQUARA – SP 2019**

Sumário

[1 Objetivos e Motivações 3](#_Toc515921877)

[1.1 Geral 3](#_Toc515921878)

[1.2 Específicos 3](#_Toc515921879)

[1.3 Motivações 4](#_Toc515921880)

[3 Fundamentos da comunicação de dados em rede 5](#_Toc515921881)

[3.1 Tipos de Redes (extensão geográfica) 5](#_Toc515921882)

[3.2 Topologia de Rede 6](#_Toc515921883)

[3.3 Transmissão de dados 7](#_Toc515921884)

[3.4 Meios de Transmissão 7](#_Toc515921885)

[3.5 Modelo OSI e TCP/IP 8](#_Toc515921886)

[3.6 Protocolos 9](#_Toc515921887)

[4 Plano de desenvolvimento da aplicação 11](#_Toc515921888)

[5 Projeto (estrutura) do programa 11](#_Toc515921889)

[6 Relatório com as Linhas de código 11](#_Toc515921890)

[7 Apresentação do programa em um computador 11](#_Toc515921891)

[Bibliografia: 11](#_Toc515921892)

# OBJETIVOS E MOTIVAÇÕES

## Geral

Abordar e apresentar uma ferramenta de comunicação em rede em multiplataforma utilizando primitivas dos sockets de Berkeley e suas derivações, sendo possível a comunicação entre os usuários através da rede com os protocolos TCP/IP. A aplicabilidade da ferramenta, é para que diversos usuários possam se comunicar para fazer uma reciclagem do seu lixo eletrônico, atualmente o número de produtos eletrônicos vem aumentando, e como a troca para um produto novo muitas vezes é necessário o conserto e para melhorar a busca por peças ou produtos que não estão sendo utilizados, desenvolvemos uma aplicação em pra ajudar na troca e venda de produtos eletrônicos usados, facilitando a reutilização.

## Específicos

• Desenvolver uma aplicação que posso fazer comunicação entre dois ou mais usuários.

• Implementar na aplicação o protocolo TCP/IP para comunicação e troca de informações entre os usuários.

• Utilizar a linguagem Java como desenvolvimento da aplicação e a interface de interações entre os usuários.

• Mostrará uma alternativa de reutilização de peças e produtos eletrônicos que possam ser utilizados por outras pessoas, não descartando de forma direta o produto como lixo.

## Motivações

A motivação para realização desta ferramenta é proporcionar a comunicação entre usuários que precisam solucionar o problema com os seus equipamentos eletrônicos, encontrar uma alternativa sustentável e ecológica do descarte do que não é mais necessário, podendo ainda dar a alternativa do reuso desse material por outras pessoas.

Através do protocolo de rede TCP/IP desenvolver uma aplicação em Java que proporcione uma aplicação que transmita mensagens, transferência de arquivos para um ou vários, fazendo a interações através das interfaces gráficas e possibilitando que a troca de informações possa ser feita pra um ou vários destinatários.

## Principais Características do Sistema

* **Linguagem escolhida:** O grupo decidiu pela utilização da linguagem Java por conta da facilidade de trabalhar com a mesma, visto que a vimos com mais frequência na faculdade do que outras, e também por conta das aulas de “Desenvolvimento de Sistemas Distribuídos” ser utilizado a mesma linguagem para a criação dos exemplos.  
  Outro motivo é também por conta do conteúdo encontrado com facilidade na internet, dessa forma, é possível tirar dúvidas frequentes de uma forma mais simples.
* **Protocolo de Comunicação:** Foi escolhido o protocolo TCP, já que o sistema se trata de um chat, no qual é necessário ter segurança no envio das mensagens, ou seja, não podem ser perdidos os dados informados por cada host na comunicação e não é necessária tanta velocidade para que o sistema possua um bom desempenho.

# INTRODUÇÃO

Com o elevado uso de equipamentos eletrônicos no mundo moderno, este tipo resíduo tem se tornado um grande problema ambiental quando não descartado em locais adequados. A sociedade precisa evitar a poluição ambiental deste resíduo, pois ele é tóxico e prejudicial à saúde dos seres vivos. Na atualidade fala se muito sobre a reciclagem de vários materiais, mas não dos equipamentos de tecnologia, que muitas vezes são prejudiciais à saúde. A cada ano o número de equipamentos vem aumentando, novas tecnologias surgem, e com isso outras tecnologia deixam de existir, ou seus fabricantes optam por não continuar a linha de peças. O que fazer se você possui um equipamento que não tem mais peças novas no mercado? Comprar um produto novo? Comprar uma peça que não é do fabricante? Ou se você tem um equipamento e troca por um atual, você descarta o que você pagou?

A resposta pra essas perguntas encontramos no desenvolvimento de uma aplicação, onde você pode anunciar alguma peça ou equipamento usado, pra vender trocar ou até mesmo doar, assim reutilizamos o material e ao invés de descartamos tudo o que não está servindo, procuramos pessoas que possam utilizar do que não nos serve.

A ideia é a publicação de anúncios, onde você pode fazer uma procura pelo que está precisando, ou divulgar o seu material. Se você se interessar por algo basta você clicar no anuncio e iniciara uma conversa com o anunciante, você podem trocar mensagens, imagens e fazer toda a negociação pela aplicação.

# FUNDAMENTOS DA COMUNICAÇÃO DE DADOS EM REDE

Uma rede de computador em termos práticos, é a ligação de dois ou mais computadores onde eles trocam recursos físicos e lógicos, esses recursos são impressoras, mensagens dados, entre outros.

O objetivo das redes de computadores é permitir troca de dados entre computadores e a partilha de hardware e software.

Antes os computadores eram conectados em curtas distancias, sendo chamado as conexões entre os dispositivos de rede locais, com o avanço e o aumento de dispositivos houve a necessidade de conexão em longas distancias para troca de informações.

## Tipos de Redes (extensão geográfica)

Redes Pessoais se comunicam entre dois ou mais dispositivos em uma curta de distância de 1,00 metro exemplo Bluetooth

Redes Locais (LAN), redes entre dispositivos de uma sala, prédio, campus universitário ou nossa casa.

Redes Metropolitanas (MAN) conexão em regiões metropolitanas, distância de 10Km, exemplo TV a cabo

Redes de Longas Distancias (WAN), rede que cobre uma grande área geográfica, ligam países.

Redes Interligadas, conectam as redes WAN, grandes ligações entre continentes

Redes sem fio, são capazes de conectar os dispositivos sem a utilização de cabeamento.

## Topologia de Rede

A topologia de redes é o padrão que os computadores e outros dispositivos estão conectados na rede, pode ser descrita do tipo física e logica.

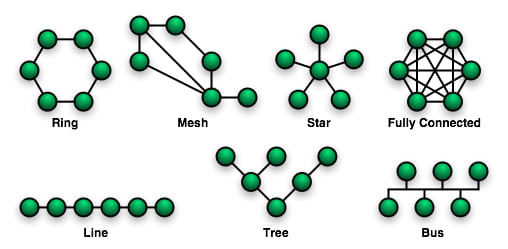
A topologia física é o layout da rede, representa como a rede está conectada e o meio de conexão dos dispositivos (nós). A forma como os cabos estão conectados influencia em pontos críticos como flexibilidade velocidade e segurança.

A topologia logica, é a forma que os dados são transmitidos na rede entre os dispositivos, descreve o fluxo dos dados através da rede.

Tipos de Topologia Física:

* Barramento: todos os dispositivos estão ligados em um mesmo barramento físico de dados.
* Estrela: O mais utilizado atualmente utiliza-se um concentrador do ponto central de rede. Os dados passam pelo concentrador que fica responsável por estabelecer uma conexão entre o dispositivo remetente e o destinatário. Essa configuração garante que se um nó estiver com problemas apenas o nó ligada não terá conexão os outros pontos ligados ao concentrador continuarão se comunicando.
* Arvore: é equivalente a várias topologias estrelas ligadas, seus concentradores são os nós centrais.
* Anel: os dispositivos são conectados em série formando um anel, uma informação enviada passara por todos os dispositivos até der retirada pelo destino ou a fonte.
* Malha: utilizada em redes mais simples, pois permite a comunicação de todos os n´s pois ficam entrelaçados, garante vários caminhos entre a origem e o destino.

A imagem abaixo da uma ilustração das topologias de rede.



Tipos de Topologia Logica:

* Broadcast: a origem envia seus dados para todos os nós destinatários da rede.
* Ethernet: O mais utilizado atualmente, quando dispositivo quer se comunicar ela precisa saber se há uma transmissão acontecendo, caso não há nenhuma transmissão ele envia seu pacote de dados ao destinatário. Se duas estações transmitem pacotes ao mesmo tempo eles serão alertados sobre o conflito e esperarão um período aleatório para transmitir novamente.
* Token Ring: utilizada na topologia em anel, a origem precisa emitir um sinal (Token) para poder percorrer os nós na rede.

## Transmissão de dados

Existe três tipos de transmissão de dados:

Simplex: um dispositivo transmissor e outro receptor , unidirecional.

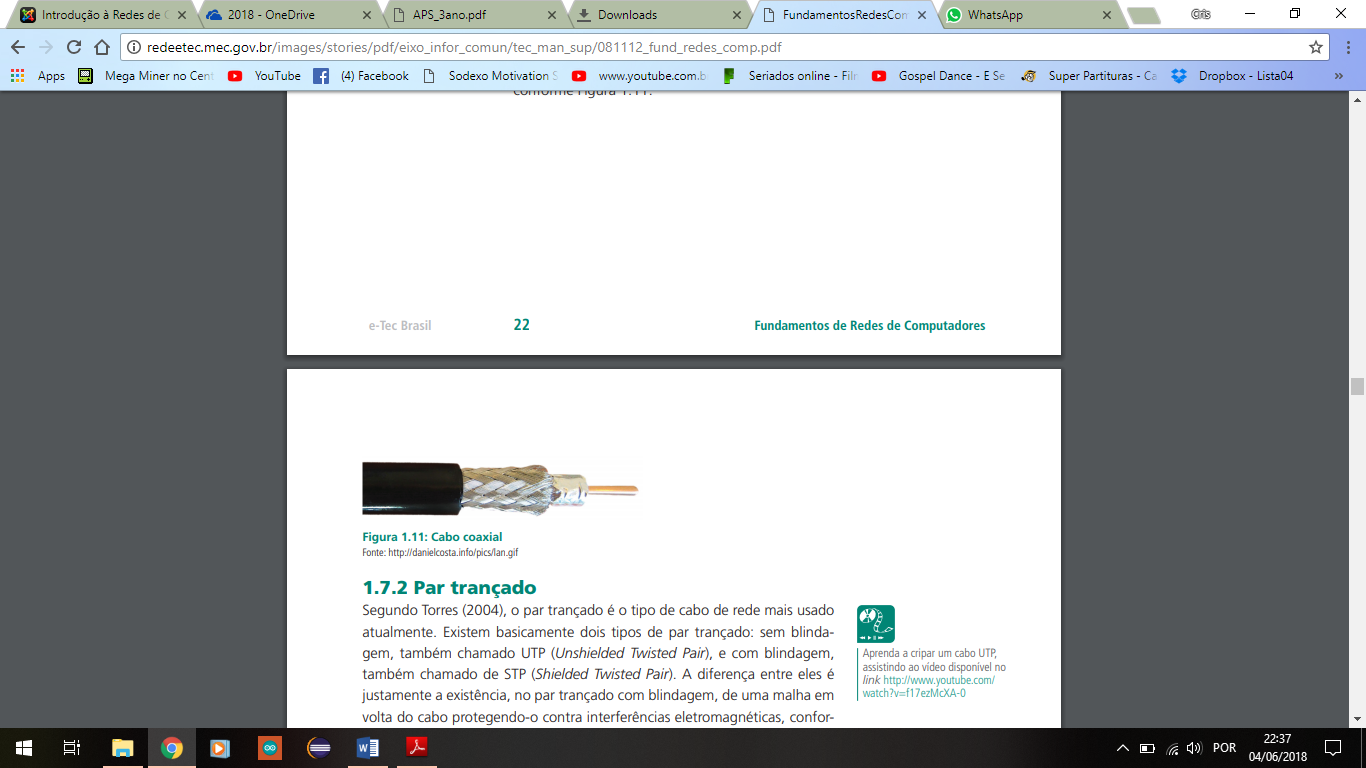
Half-Duplex: bidirecional, enviam os dados pelo mesmo canal de comunicação, não faz envio e recebimento ao mesmo tempo.

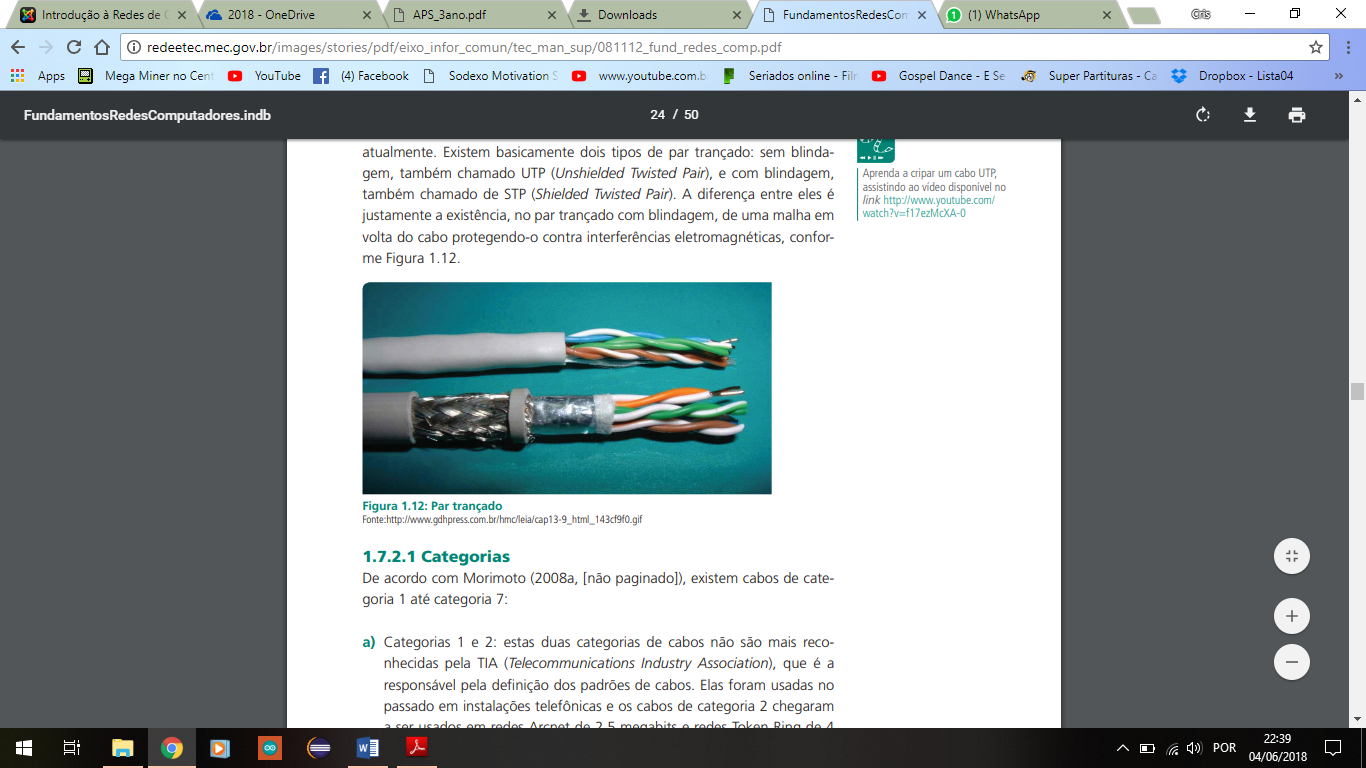
Full-Duplex: bidirecional, como uma rodovia de pista dupla, um dado é enviado quanto outro pode estar sendo recebido.

## Meios de Transmissão

São os meios físicos utilizados para transmitirem as informações. Eles possuem características próprias como largura de banda, retardo, custo e facilidade de instalação e manutenção. São agrupados em meios guiados e não guiados.

Guiados fio de cobre e fibra-ótica, Não-Guiados ondas de rádio e raio laser transmitidos pelo ar.

 Cabo Coaxial, fio de cobre com na parte central com envolvido por uma material isolante.

 Cabo Par Trançado, o mais utilizado atualmente, temos o tipo com blindagem(STP) e sem blindagem(UTP), são divido em categorias que vão de 1 a 7 que diferenciam na velocidade transmissão dos dados que alcançam.

Fibra Ótica, transmite informações através de sinais luminosos, é imune a ruídos e a comunicação é superior ao par trançado. São classificados de acordo com o trafego da luz no cabo, monomodo, luz é transmitida de forma direta no núcleo do cabo, atinge distancia maiores, no multimodo a luz faz diferentes refrações no trajeto.

## Modelo OSI e TCP/IP

Para que diversos dispositivos se comunicassem foi necessário a criação de um padrão para que independente do fabricante e da tecnologia utilizada os dispositivos pudessem se comunicar. Nesse contexto foi criado o modelo OSI em 1984 pela ISO. É apenas um conceito que descreve como deve ser feita a comunicação de dados entre os equipamentos é o modelo de referência para todos os protocolos de comunicação e é dividido em camadas para facilitar o entendimento e a detecção de problemas.

O Modelo OSI é dividido em 7 camadas:

1º. Física (Assegurar o transporte de bits através de um meio de transmissão).

2º.Enlace (Transmissão de quadros entre duas maquinas, envia os quadros sequencialmente, regula o trafego, detecta erros da camada física)

3º Rede (Encaminhar os pacotes entre a origem e o destino, faz o roteamento, controla o congestionamento, tamanho do pacote)

O trabalho é desenvolvido para trabalhar nessa camada, utilizamos Socket e ServerSocket que são encontrados na biblioteca java.io.\*

A função dessas classes é trabalhar na terceira camada, utilizando o endereço de IP e a Portas.

4º Transporte (Efetuar a comunicação fim-a-fim entre os processos, garante a entrega da mensagem na ordem correta, recebe os dados enviados da camada de sessão e segmenta para enviar pra camada de redes.)

5º. Sessão (controla quem fala e quando fala, entre a origem e o destino).

6º. Apresentação (Converte o formato dos dados recebidos pela camada de aplicação em um formato para transmissão dos dados, um formato entendido pelo protocolo)

7º Aplicação (Programas, fazem a interação entre a máquina e o usuário).

Os conceitos fundamentas da camada OSI são Serviço que informa o que a camada faz, Interface informa como os processos podem acessa-los e Protocolo a camada pode utilizar o protocolo que desejar desde que este realize o trabalho.

O modelo TCP/IP (Transmisson Control Protocol / Internet Protocol) é o mais conhecido, ele é divido em 4 camadas.

1º. Acesso à rede (Interface entre o dispositivo e o enlace de transmissão)

2º. Internet (Protocolo IP (Internet Protocol), faz com que os pacotes enviados cheguem ao seu destino independentes de falhas e perdas de pacotes).

3º.Transporte (Permite que os dispositivos de origem e destinos conversem independente da distância, utilizam o protocolo TCP (Transmission Control Protocol) orientado a conexão confiáveis e UDP (User Datagran Protocol) sem conexões, não confiáveis.

4º. Aplicação (Protocolos de alto nível Telnet, FTP, SMTP, DNS, HTPP, etc.)

## Protocolos

Protocolos é o padrão estabelecido para que os dispositivos consigam se comunicar, é o conjunto de regras que governa a comunicação de dados.

Principais protocolos:

* IP (INTERNET PROTOCOL ou PROTOCOLO DE INTERNET)

É o protocolo pelo qual os dados são enviados a partir de um computador para outro na Internet. É responsável pelo envio e endereçamentos dos pacotes TCP.

* TCP/IP (PROTOCOLO DE CONTROLE DE TRANSMISSÃO / PROTOCOLO INTERNET)

Forma padronizada de comunicação para os computadores na internet. O TCP faz a divisão dos dados que será enviada em segmentos de dados.

* HTTP (HYPERTEXT TRANSFER PROTOCOL / PROTOCOLO DE TRANSFERÊNCIA DE HIPERTEXTO).

Usado para fazer o acesso à WWW (WORLD WIDE WEB) como protocolo de cliente/servidor. Se na barra de endereços do navegador não se especificar outro recurso, o este protocolo será tido como padrão.

* FTP (FILE TRANSFER PROTOCOL / PROTOCOLO DE TRANSFERÊNCIA DE ARQUIVO)

Recurso utilizado para transferências de arquivos pela internet. Este protocolo permite apenas o envio e o recebimento dos arquivos, no entanto para a leitura serão necessários outros programas.

* SSL (SECURE SOCKETS LAYER / CAMADA DE PORTAS DE SEGURANÇA)

Este é um protocolo de segurança que permite a confirmação da identidade de um servidor, verificando o nível de confiança.

* ICMP (INTERNET CONTROL MESSAGE PROTOCOL / PROTOCOLO DE MENSAGENS DE CONTROLE DA INTERNET)

Autoriza a criação de mensagens relativas ao IP, mensagens de erro e pacotes de teste.

* SMTP (SIMPLE MAIL TRANSFER PROTOCOL / PROTOCOLO PARA TRANSFERÊNCIA DE E-MAIL SIMPLES)

Usado para envios de mensagens em rede funcionando como roteador do correio eletrônico.

* IMAP (INTERNET MESSAGE ACCESS PROTOCOL / PROTOCOLO DE ACESSO AO CORREIO DA INTERNET)

Permite a manipulação de caixas postais remotas como se fossem locais, permitindo a organização da forma que melhor convier.

* POP (POST OFFICE PROTOCOL ou PROTOCOLO DOS CORREIOS)

Usado para receber mensagens de e-mail. Permitem apenas o acesso a uma caixa de e-mail, já o IMAP permite o acesso a várias.

* MIME (MULTIPURPOSE INTERNET MAIL EXTENSIONS/EXTENSÕES MULTIFUNÇÃO PARA MENSAGENS DE INTERNET)

Permite ao usuário a utilização do e-mail trabalhando em multimídia.

# PLANO DE DESENVOLVIMENTO DA APLICAÇÃO

O desenvolvimento da aplicação se inicia com a construção do servidor central, estabelece a conexão entre os usuários, faz a transmissão dos dados e estabelece o relacionamento entre os usuários.

A aplicação pode ser iniciada após a conexão do servidor ser estabelecida.

O desenvolvimento de uma interface onde filtra os usuários precisam fornecer seus dados para verificação no banco de dados, essa etapa é pra deixar salvo os contatos e para reconhecimento do usuário dentro da aplicação.

Inserimos um campo onde novos usuários possam se cadastrar e ter acesso ao sistema.

No cadastro o usuário precisa fornecer seu nome, um e-mail e uma senha para ser inserido no sistema.

Após a autenticação do usuário é apresentada a tela principal, onde fica todos os anúncios e os usuários que já estão conectados à rede.

Implementados uma interface onde cada as pessoas conectadas possam publicar seus anúncios do material que querem descartar, ele precisa inserir uma imagem, um título e uma descrição e pode publicar para que todos os outros usuários vejam seu anuncio.

Se houver o interesse pela publicação, implementamos uma função onde ao clicar no anuncio inicia uma conversa em forma de chat, nessa interface os usuários podem trocar mensagens e arquivos.

# PROJETO (ESTRUTURA) DO PROGRAMA

O projeto ao todo foi subdivido em:

* Aplicação;

A aplicação é responsável pela interação com o usuários e também é dividia em interface, processamentos dos dados do usuário, comunicação com o servidor.

* Servidor;

Servidor é responsável pela conexão entre usuários, também pelo tráfegos dos dados, validação das informações, é dividido em servidor central, servidor de autenticação, servidor de chat.

# 6 RELATÓRIO COM AS LINHAS DE CÓDIGO

ServidorCentral.java

1 package ecochat.aplicacoes.servidor;

23

import java.io.IOException;

18

19 public class ServidorCentral {

20

21 private static ServerSocket socketServidorCentral;

22 private static List<Socket> socketsConectados;

23 private static ServidorCentral instancia;

24 public Socket socketAnuncio;

25

26 public static void main(String[] args) {

27

28 try {

29 UIJanelaServidorCentral.getInstance();

30 } catch (HibernateException exception) {

31 exception.printStackTrace();

32 } catch (Exception e) {

33 System.err.println("Ops! " + e.getMessage() + "\n");

34 }

35 }

36

37 public void iniciarServidor() {

38

39 try {

40 ServidorAutenticacao.getInstance().iniciarServidor();

41 ServidorChat.getInstance().iniciarServidor();

42

43 socketServidorCentral = new

ServerSocket(ConstantesGerais.PORTA\_SERVIDOR\_CENTRAL,

44 ConstantesGerais.QUANTIDADE\_MAXIMA\_CONECTADOS,

45 InetAddress.getByName(ConstantesGerais.IP\_SERVIDOR\_CENTRAL));

46

47 socketsConectados = new ArrayList<Socket>();

48 UIJanelaServidorCentral.getInstance().mostrarMensagem(" ---=====

Servidor Conectado =====---");

49 while (true) {

50

51 Socket socket = socketServidorCentral.accept();

52 if

(socket.getInetAddress().getHostAddress().equals(ConstantesGerais.IP\_FIXO\_ENVIO\_ANUNCIO)) {

53 socketAnuncio = socket;

54 atualizarPaineis();

55 } else {

56

57 UIJanelaServidorCentral.getInstance().mostrarConectados(socket.getInetA

ddress().getHostAddress());

58

59 socketsConectados.add(socket);

60

61 atualizarUsuariosOnlines(socket.getInetAddress().getHostAddress());

62 }

63 }

64 } catch (IOException ioE) {

65 System.err.println(ioE.getMessage());

66 }

67 }

68

69 public void notificarUsuario(final DadoCompartilhado dadoCompartilhado) {

70 new Thread() {

71 public void run() {

72

Page 1

ServidorCentral.java

73 for (Socket socketConectado : socketsConectados) {

74 String ipSocketConectado =

socketConectado.getInetAddress().getHostAddress();

75 if (ipSocketConectado.equals(dadoCompartilhado.getDestinatario())) {

76 ObjectOutputStream fluxoSaidaDados;

77 try {

78 fluxoSaidaDados = new

ObjectOutputStream(socketConectado.getOutputStream());

79

80 DadoCompartilhadoServidor dadoCompartilhadoServidor = new

DadoCompartilhadoServidor();

81 dadoCompartilhadoServidor.setDadoCompartilhado(dadoCompartilhad

o);

82

83 fluxoSaidaDados.writeObject(dadoCompartilhadoServidor);

84 } catch (IOException e) {

85 e.printStackTrace();

86 }

87 }

88 }

89 };

90 }.start();

91 }

92

93 public void desligarServidor() {

94 try {

95

96 for (Socket socketConectado : socketsConectados) {

97 socketConectado.close();

98 }

99

100 socketServidorCentral.close();

101 UIJanelaServidorCentral.getInstance().mostrarMensagem(" ---===== Servidor

Desconectado =====---");

102

103 } catch (IOException ioE) {

104 System.err.println("Falha ao desligar o servidor\n\n" + ioE.getMessage());

105 }

106 }

107

108 public void atualizarUsuariosOnlines(final String ipSocketConectado) {

109 new Thread() {

110 public void run() {

111 List<Socket> socketsConectadosCopia = new

ArrayList<Socket>(socketsConectados);

112 try {

113

114 Socket socketSeraAtualizado = null;

115

116 for (Socket socketConectado : socketsConectadosCopia) {

117

118 String ipSocketLista =

socketConectado.getInetAddress().getHostAddress();

119 if (!ipSocketLista.equals(ipSocketConectado)) {

120 ObjectOutputStream fluxoSaidaDados = new ObjectOutputStream(

121 socketConectado.getOutputStream());

122

123 DadoCompartilhadoServidor dadoCompartilhadoServidor = new

DadoCompartilhadoServidor();

124 dadoCompartilhadoServidor.setIpUsuarioConectou(ipSocketConectad

o);

125

Page 2

ServidorCentral.java

126 fluxoSaidaDados.writeObject(dadoCompartilhadoServidor);

127 } else

128 socketSeraAtualizado = socketConectado;

129 }

130

131 for (Socket socketConectado : socketsConectadosCopia) {

132

133 String ipSocketLista =

socketConectado.getInetAddress().getHostAddress();

134 if (!ipSocketLista.equals(ipSocketConectado)) {

135 ObjectOutputStream fluxoSaidaDados = new ObjectOutputStream(

136 socketSeraAtualizado.getOutputStream());

137

138 DadoCompartilhadoServidor dadoCompartilhadoServidor = new

DadoCompartilhadoServidor();

139 dadoCompartilhadoServidor.setIpUsuarioConectou(ipSocketLista);

140

141 fluxoSaidaDados.writeObject(dadoCompartilhadoServidor);

142 }

143 }

144

145 } catch (Exception e) {

146 e.printStackTrace();

147 }

148 }

149 }.start();

150 }

151

152 private void atualizarPaineis() {

153 new Thread() {

154 public void run() {

155 while (true) {

156 try {

157 ObjectInputStream fluxoEntradaDados = new

ObjectInputStream(socketAnuncio.getInputStream());

158 DadoCompartilhadoServidor dcServidor = (DadoCompartilhadoServidor)

fluxoEntradaDados

159 .readObject();

160

161 List<Socket> socketsConectadosCopia = new

ArrayList<Socket>(socketsConectados);

162 for (Socket socketConectado : socketsConectadosCopia) {

163 ObjectOutputStream fluxoSaidaDados = new ObjectOutputStream(

164 socketConectado.getOutputStream());

165 fluxoSaidaDados.writeObject(dcServidor);

166 }

167

168 } catch (Exception e) {

169 e.printStackTrace();

170 }

171 }

172 }

173 }.start();

174 }

175

176 public static ServidorCentral getInstance() {

177

178 if (instancia == null)

179 return instancia = new ServidorCentral();

180

181 return instancia;

182 }

Page 3

ServidorCentral.java

183 }

184

Page 4

ControleChatAplicacao.java

1 package ecochat.aplicacoes.servidor.controle;

23

import java.io.File;

15

16 public class ControleChatAplicacao {

17

18 private static ControleChatAplicacao instancia;

19

20 private ControleChatAplicacao() {

21 UIJanelaChat.getInstance();

22 lerMensagemServidor();

23 }

24

25 public void enviarMensagemAoServidor(final DadoCompartilhado dadoCompartilhado) {

26

27 new Thread() {

28 public void run() {

29 try {

30

31 if (dadoCompartilhado.getArquivo() != null) {

32 Thread.sleep(2000);

33 UIJanelaChat.getInstance().trocarLoadingPorImagemArquivo("Você

Enviou",

34 dadoCompartilhado.getArquivo());

35 }

36

37 ObjectOutputStream fluxoSaidaDados =

ControlePainelPrincipalAnuncios.getInstance()

38 .getFluxoSaidaDados();

39 dadoCompartilhado.setRemetente(ControlePainelPrincipalAnuncios.getInsta

nce().getIpAplicacao());

40 fluxoSaidaDados.writeObject(dadoCompartilhado);

41 fluxoSaidaDados.flush();

42 } catch (IOException e) {

43 e.printStackTrace();

44 } catch (InterruptedException e) {

45 e.printStackTrace();

46 }

47 }

48 }.start();

49 }

50

51 private void lerMensagemServidor() {

52 new Thread() {

53 @SuppressWarnings("static-access")

54 public void run() {

55

56 try {

57 while (true) {

58

59 ObjectInputStream fluxoEntradaDados = new ObjectInputStream(

60 ControlePainelPrincipalAnuncios.getInstance().getSocket().g

etInputStream());

61

62 Object leituraObjeto = fluxoEntradaDados.readObject();

63 if (UIJanelaChat.getInstance().isFocusableWindow()) {

64 UIJanelaChat.setMensagemNaFila(false);

65 }

66

67 DadoCompartilhado dadoCompartilhado = (DadoCompartilhado)

leituraObjeto;

68 UIJanelaChat.getInstance().receberMensagem(dadoCompartilhado.getMen

Page 1

ControleChatAplicacao.java

sagem());

69

70 if (dadoCompartilhado.getArquivo() != null) {

71

72 UIJanelaChat.getInstance().adicionarAnimacaoArquivo();

73

74 InputStream entradaArquivo = null;

75 OutputStream saidaArquivo = null;

76

77 try {

78 File arquivo = dadoCompartilhado.getArquivo();

79

80 arquivo.createTempFile(Utilitaria.recuperarPastaDownload(),

"");

81 entradaArquivo = new FileInputStream(arquivo);

82 saidaArquivo = new FileOutputStream(

83 new File(Utilitaria.recuperarPastaDownload() +

Utilitaria.gerarNomeArquivo()

84 +

Utilitaria.recuperarExtensaoArquivo(arquivo)));

85

86 this.sleep(2000);

87

88 byte[] memoriaTemporaria = new byte[1024 \* 50];

89 int tamanho;

90 while ((tamanho = entradaArquivo.read(memoriaTemporaria)) >

0) {

91 saidaArquivo.write(memoriaTemporaria, 0, tamanho);

92 }

93

94 UIJanelaChat.getInstance().trocarLoadingPorImagemArquivo("V

ocê Recebeu",

95 dadoCompartilhado.getArquivo());

96 } catch (Exception ex) {

97 System.err.println(ex.getMessage());

98 } finally {

99

100 entradaArquivo.close();

101 saidaArquivo.flush();

102 saidaArquivo.close();

103 }

104 }

105 }

106 } catch (IOException | ClassNotFoundException e) {

107 e.printStackTrace();

108 System.err.println(e.getMessage());

109 } catch (Exception ex) {

110 System.err.println(ex.getMessage());

111 }

112 }

113 }.start();

114 }

115

116 public static ControleChatAplicacao getInstance() {

117

118 if (instancia == null)

119 return instancia = new ControleChatAplicacao();

120

121 return instancia;

122 }

123 }

ControlePainelPrincipalAnuncios.java

1 package ecochat.aplicacoes.servidor.controle;

23

import java.io.IOException;

14

15 public class ControlePainelPrincipalAnuncios {

16

17 private static Socket socketServidorCentral;

18 private static Socket socketServidorChat;

19 private static ObjectOutputStream fluxoSaidaDados;

20 private static ControlePainelPrincipalAnuncios instancia;

21 private String ipMaquina;

22

23 private ControlePainelPrincipalAnuncios() {

24 try {

25 ipMaquina = Utilitaria.criarIpAleatorio();

26 UIJanelaPrincipal.getInstance(ipMaquina);

27 conectarServidor();

28 iniciarLeituraAtualizacoesSistema();

29 } catch (UnknownHostException e) {

30 e.printStackTrace();

31 } catch (IOException e) {

32 e.printStackTrace();

33 } catch (InterruptedException e) {

34 e.printStackTrace();

35 }

36 }

37

38 private void conectarServidor() throws UnknownHostException, IOException,

InterruptedException {

39

40 socketServidorCentral = new

Socket(InetAddress.getByName(ConstantesGerais.IP\_SERVIDOR\_CENTRAL),

41 ConstantesGerais.PORTA\_SERVIDOR\_CENTRAL, InetAddress.getByName(ipMaquina),

0);

42

43 Thread.sleep(1000);

44

45 socketServidorChat = new

Socket(InetAddress.getByName(ConstantesGerais.IP\_SERVIDOR\_CHAT),

46 ConstantesGerais.PORTA\_SERVIDOR\_CHAT, InetAddress.getByName(ipMaquina), 0);

47

48 fluxoSaidaDados = new ObjectOutputStream(socketServidorChat.getOutputStream());

49 fluxoSaidaDados.flush();

50 }

51

52 private void iniciarLeituraAtualizacoesSistema() {

53

54 new Thread() {

55 public void run() {

56 while (true) {

57 try {

58

59 ObjectInputStream fluxoEntradaDados = new ObjectInputStream(

60 socketServidorCentral.getInputStream());

61 DadoCompartilhadoServidor dadoCompartilhadoServidor =

(DadoCompartilhadoServidor) fluxoEntradaDados

62 .readObject();

63

64 String ipUsuarioConectou =

dadoCompartilhadoServidor.getIpUsuarioConectou();

65

66 if (ipUsuarioConectou != null)

Page 1

ControlePainelPrincipalAnuncios.java

67 UIJanelaPrincipal.getInstance().adicionarUsuariosOnline(ipUsuar

ioConectou);

68

69 // TODO FAZER ENVIO DE ANÚNCIOS

70 if(dadoCompartilhadoServidor.getAnuncio() != null) {

71 UIJanelaPrincipal.getInstance().adicionaPainel(dadoCompartilhad

oServidor.getAnuncio());

72 }

73

74 if (dadoCompartilhadoServidor.getDadoCompartilhado() != null)

75 UIJanelaPrincipal.getInstance()

76 .notificarUsuario(dadoCompartilhadoServidor

77 .getDadoCompartilhado().getRemetente());

78

79 } catch (ClassNotFoundException e) {

80 e.printStackTrace();

81 } catch (Exception ex) {

82 ex.printStackTrace();

83 }

84 }

85 }

86 }.start();

87 }

88

89 public String getIpAplicacao() {

90 return ipMaquina;

91 }

92

93 public Socket getSocket() {

94 return socketServidorChat;

95 }

96

97 public ObjectOutputStream getFluxoSaidaDados() {

98 return fluxoSaidaDados;

99 }

100

101 public static ControlePainelPrincipalAnuncios getInstance() {

102

103 if (instancia == null)

104 return instancia = new ControlePainelPrincipalAnuncios();

105

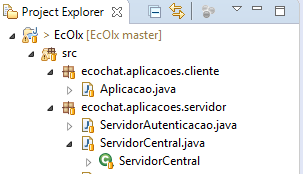
106 return instancia;

107 }

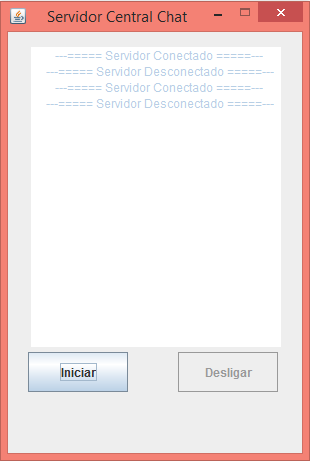
108 }

# 7 APRESENTAÇÃO DO PROGRAMA EM UM COMPUTADOR

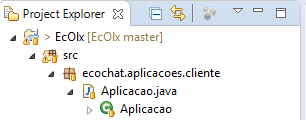
Para iniciar a aplicação é necessária executar a classe “ServidorCentral” (banco de dados) para que ele possa fazer as consultas dos usuários cadastrados e realizar a inclusão dos novos usuários.



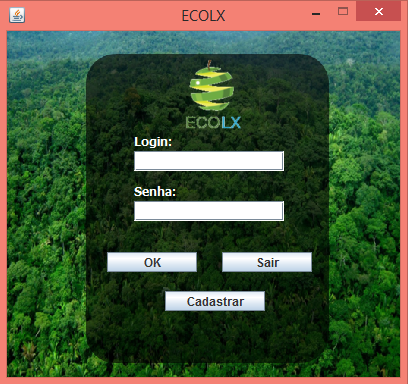
Uma nova janela será apresentada com os botões “Iniciar”, que ao selecionarmos, inicializa o banco de dados para que faça o controle de inserções de usuários e o botão “Desligar”, para encerrar a aplicação do banco de dados. A imagem abaixo ilustra o funcionamento dos botões “Iniciar” e “Desligar”, respectivamente.

Com o “ServidorCentral” em funcionamento já é possível rodar a aplicação do programa, através da classe “Aplicação”.

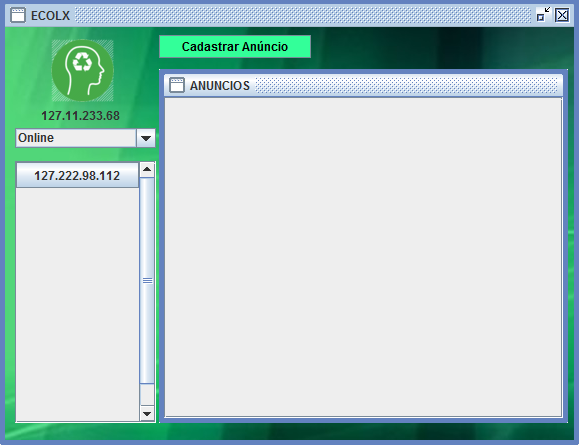


Será apresentada a interface de login ao usuário, caso deste ser um usuário cadastrado o mesmo deverá inserir seu login e senha, já nos casos dos mesmos serem novos usuários, será direcionado para uma nova janela para a criação do cadastro do usuário.



Interface de login apresenta ao usuário os botões para acesso ao programa, através do botão ok (após o usuário preencher corretamente os campos login e senha). O botão sair deve ser utilizado quando o usuário desejar fechar a aplicação.

Após a inserção das informações na tela de login, o usuário será direcionado a tela principal do aplicativo, lá terá acesso as informações dos usuários que estão on-line, aos anúncios disponíveis e terá também a liberdade de criar novos anúncios no “ECOLX”.



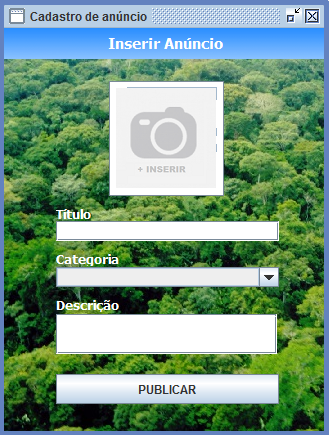
4

3

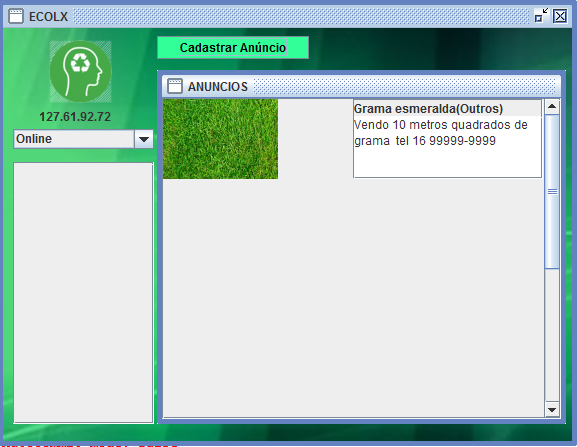
2

1

A imagem acima foi dividida e enumerada em blocos para facilitar o entendimento, o primeiro bloco mostra informações sobre o próprio usuário e o box de status do mesmo (Online, Offline, Ausente e Ocupado) que pode ser alterado. O segundo bloco está destinado a apresentação dos usuários online no aplicativo e lá o usuário poderá fazer contato com outros usuários que também estejam online. Ao clicar no bloco com o IP será aberta uma nova janela de mensagens onde ambos podem trocar mensagens e enviar imagens entre si. No terceiro bloco são exibidos todos os anúncios criados e no bloco 4 o usuário é direcionado a janela de criação de anúncio, demonstrada abaixo.



Após a publicação, o anúncio será exibido na tela principal com as informações preenchidas.



O anuncio será exibido ao clicar em publicar.

# BIBLIOGRAFIA

Tanenbaum, Andrew S. - Redes de Computadores - Rio de Janeiro : Campus, 2003. Através do site <http://www.teraits.com/pitagoras/marcio/gpi/b_ATanenbaum_RedesDeComputadores_4aEd.pdf>

Acessado em 01.05.2018

DEITEL & DEITEL; Java – Como Programar, 4ª Edição; Editora Bookman.

Através do site

<http://www.koflinux.com.br/java.pdf>

Acessado em 06.05.2018