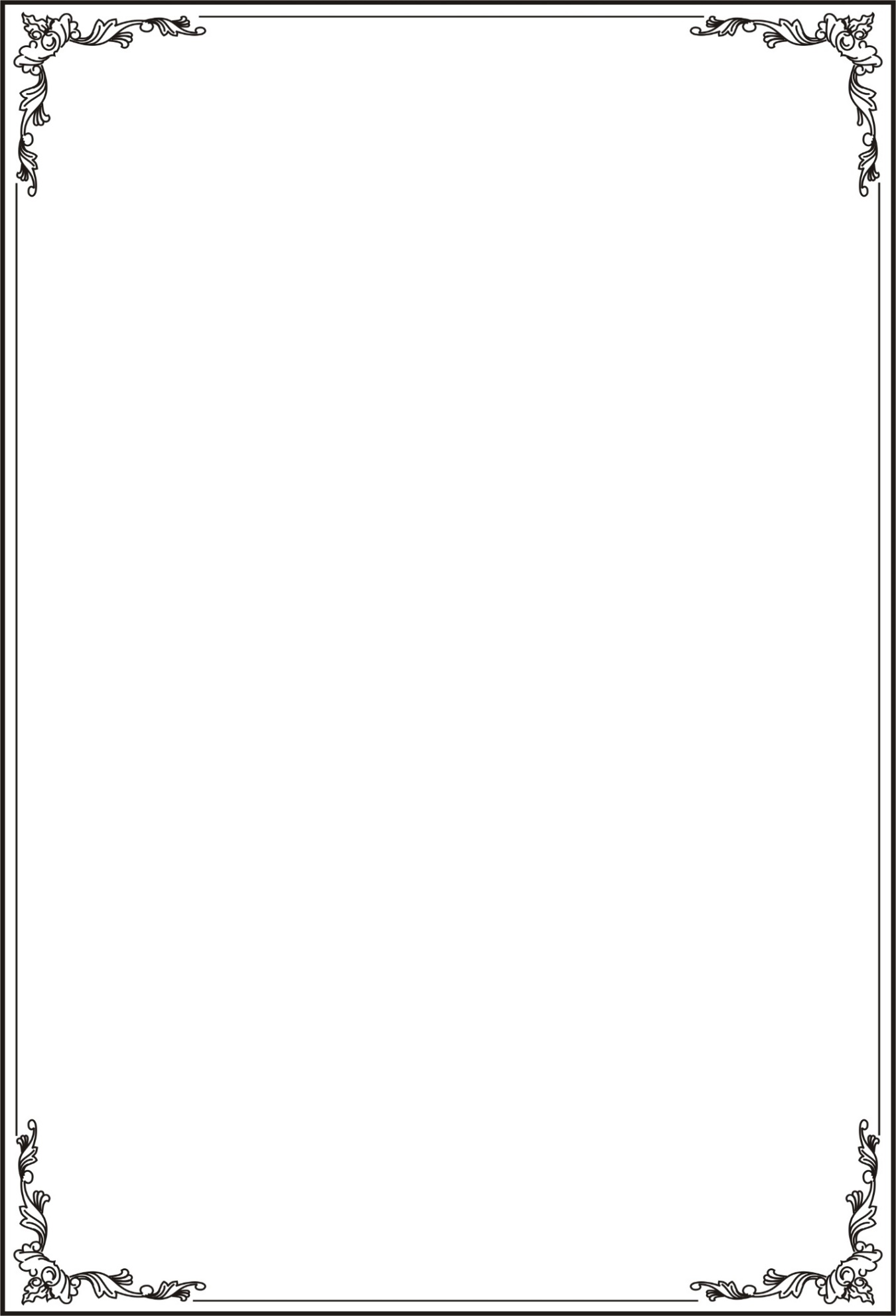
****

**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**- - - 🙞 🕮** **🙜 - - -**



**LẬP TRÌNH MẠNG**

Đề tài: Xây dựng chương trình IRC Client

Giáo viên hướng dẫn: **ThS. Bùi Trọng Tùng**

Sinh viên thực hiện: MSSV

Đinh Hoàng Hải Hảo 20141412

Hà Nội, 16/06/2017

**MỤC LỤC**

[**LỜI MỞ ĐẦU** 4](#_Toc485237658)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 4](#_Toc485237659)

[**CHƯƠNG 1: TỔNG QUAN VỀ INTERNET RELAY CHAT** 5](#_Toc485237660)

[**1. Các khái niệm cơ bản** 5](#_Toc485237661)

[**2. Những quy định về message** 5](#_Toc485237662)

[2.1. Dạng chung của thông điệp: 5](#_Toc485237663)

[2.2. Dạng chung của message: 5](#_Toc485237664)

[**3. Chi tiết từng loại Message (Message detail)** 5](#_Toc485237665)

[3.1. Nhóm message đăng ký kết nối: client gửi thông tin lên server để kết nối tới server 5](#_Toc485237666)

[3.2. Nhóm message dùng điều khiển channel: 5](#_Toc485237667)

[3.3. Nhóm message truy vấn đến server (server query and command): 6](#_Toc485237668)

[3.4. Nhóm message gửi text 6](#_Toc485237669)

[3.5. Nhóm message do client truy vấn đến server (user-based query) 6](#_Toc485237670)

[3.6. Nhóm message khác (miscellaneous message) 6](#_Toc485237671)

[3.7. Nhóm message tùy chọn (option message) 7](#_Toc485237672)

[**CHƯƠNG 2: MÔ TẢ GIAO THỨC INTERNET RELAY CHAT** 8](#_Toc485237673)

[**1. Mô hình Client kết nối đến Server** 8](#_Toc485237674)

[**2. Mô hình client tham gia vào channel** 9](#_Toc485237675)

[**3. Mô hình thoát khỏi channel và hủy bỏ kết nối** 9](#_Toc485237676)

[**CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH** 10](#_Toc485237677)

[**1. Thiết kế chương trình** 10](#_Toc485237678)

[1.1. Giao diện của chương trình 10](#_Toc485237679)

[1.2. Danh sách chức năng 12](#_Toc485237680)

[**2. Cấu trúc của chương trình** 12](#_Toc485237681)

[2.1. Package Interface 12](#_Toc485237682)

[2.2. Package Process 13](#_Toc485237683)

[**3. Hoạt động của chương trình** 14](#_Toc485237684)

[3.1. Hoạt động kết nối 14](#_Toc485237685)

[3.2. Hoạt động ngắt kết nối 14](#_Toc485237686)

[3.3. Hoạt động Open Chat on Channel 14](#_Toc485237687)

[3.4. Hoạt động Open Chat One 15](#_Toc485237688)

[3.5. Hoạt động của quá trình nhận message 15](#_Toc485237689)

[**CHƯƠNG 4: THỬ NGHIỆM VÀ ĐÁNH GIÁ CHƯƠNG TRÌNH** 16](#_Toc485237690)

[**1. Kiểm thử chương trình** 16](#_Toc485237691)

[1.1. Yêu cầu của chương trình 16](#_Toc485237692)

[1.2. Kịch bản kiểm thử 16](#_Toc485237693)

[**2. Đánh giá chương trình** 16](#_Toc485237694)

[2.1. Ưu điểm 16](#_Toc485237695)

[2.2. Nhược điểm 16](#_Toc485237696)

[**KẾT LUẬN** 18](#_Toc485237697)

# **LỜI MỞ ĐẦU**

Hiện nay trên Internet có nhiều loại dịch vụ, mỗi dịch vụ cung cấp cho chúng ta một tiện ích khác nhau, trong đó có dịch vụ chat. Đây là loại dịch vụ cho phép mọi người ở bất cứ đâu đều có thể gặp gỡ, trao đổi thông tin với nhau mà không cần phải gặp nhau trực tiếp. Vào những năm 80 của thế kỷ 20, dịch vụ này rất phát triển, số lượng người tham gia sử dụng ngày càng tăng. Vì vậy cần phải có một quy định chung, một giao thức chung cho dịch vụ chat trong hệ thống mạng. Năm 1980, giao thức IRC (Internet Relay Chat) ra đời và được áp dụng cho tất cả các chương trình chat đang tồn tại.

IRC được coi là một dạng liên lạc cấp tốc qua mạng Internet.Nó thiết kế với mục đích chính là cho phép các nhóm người trong một phòng thảo luận liên lạc với nhau.. Hiện nay IRC là mạng trò truyện trực tuyến lớn, có vài triệu kênh trên máy phục vụ trên khắp thế giới và có rất nhiều phầm mềm IRC phổ biến như mIRC, AdiIRC, KiwiIRC, HexChat, IceChat IRC,…

Giao thức IRC được xây dựng trên họ giao thức mạng phổ biến nhất là TCP/IP, lý do việc sử dụng họ giao thức này là tính chính xác, tin cậy, phổ biến, rất thích hợp cho các thảo luận từ xa.

Giao thức IRC sử dụng mô hình client – server, vì thế chúng ta có thể chạy nhiều máy trên môi trường phân tán. Trong đó máy đóng vai trò server cung cấp một điểm tập trung cho các client kết nối đến, và đồng thời thực hiện quá trình truyền nhận message từ các client này đến client khác.

# **DANH MỤC TÀI LIỆU THAM KHẢO**

Luận văn: Tìm hiểu và ứng dụng Internet Relay Chat Protocol

# **CHƯƠNG 1: TỔNG QUAN VỀ INTERNET RELAY CHAT**

## **1. Các khái niệm cơ bản**

- Server: được xem như là xương sống của mạng IRC, mỗi server là một tâm điểm trong hệ thống các server, chúng cho phép các client và các server khác kết nối vào.

- Client: là một máy tính mà nó được kết nối đến server và máy tính đó không phải là server, khi client kết nối đến server, server sẽ nhận được thông tin của client và client có thể sử dụng các dịch vụ chat.

- Channel: là tên nhóm một hay nhiều client, mà những client này sẽ cùng nhận được các message gửi đến channel đó, channel có thể hiều là một nhóm được tạo để những client trong nhóm có thể trao đổi với nhau.

- Truyền thông một – một (one to one communication): là chuyển thông điệp từ một client này đến một client khác thông qua một server nào đó.

- Truyền thông một – nhiều (one to many communication): là loại truyền thông có mục đích sử dụng cho các cuộc hội thảo, thảo luận,… hoặc dùng để quảng bá, quảng cáo. Có hai hình thức là One to List và One to Group.

- Truyền thông một – tất cả (one to all communication): là loại truyền thông này giống như quảng bá, trong đó message được gửi đến tất cả các client và server trên mạng và chúng sẽ tự tìm kiếm đường đi.

## **2. Những quy định về message**

### 2.1. Dạng chung của thông điệp:

- Có 2 loại thông điệp là thông điệp xuất phát từ lệnh (thông điệp mà client gửi tới server) và thông điệp trả về (thông điệp mà server gửi lại client).

- Có 2 dạng thông điệp là thông điệp bằng chữ và thông điệp bằng số.

### 2.2. Dạng chung của message:

- Message gồm 3 phần: phần đầu còn gọi là phần tiếp đầu ngữ (prefix), tiếp theo là phần lệnh (command) và cuối cùng là danh sách đối số (parameters list), mỗi danh sách cách nhau bởi 1 ký tự khoảng trắng và message được kết thúc bằng 2 ký tự “\r\n”.

## **3. Chi tiết từng loại Message (Message detail)**

### 3.1. Nhóm message đăng ký kết nối: client gửi thông tin lên server để kết nối tới server

- Client gửi Password tới server: “PASS <password> \r\n”

- Client gửi Nickname tới server: “NICK <nickname> \r\n”

- Client gửi Username tới server: “USER <username> \r\n”

- Để ngắt kết nối tới server, client gửi tới server: “QUIT :[<Quit message>]\r\n”

### 3.2. Nhóm message dùng điều khiển channel:

- Tham gia channel: JOIN <channel>{,<channel} [<key>][,<key>]

- Thoát khỏi channel: PART <channel>{,<channel>}

- Thay đổi hoặc xem chủ đề (topic) của channel: TOPIC <channel>[<topic>]

- Xem danh sách user trên channel: NAMES [<channel>]{,<channel>}

- Xem danh sách channel trên server: LIST [<channel>{,<channel>}[<server>]]

- Mời user vào channel: INVITE <nickname><channel>

- Loại bỏ user khỏi channel: KICK <channel><user>[<comment>]

### 3.3. Nhóm message truy vấn đến server (server query and command):

- Kiểm tra version chương trình của server: VERSION [<server>]

- Truy vấn thông tin từ server: STATS [<query>[<server>]]

- Liệt kê tất cả những server mà <remote server> biết:

LINKS [[<remote server>]<server mask>]

- Truy vấn thời gian từ một server nào đó: TIME [<server>]

- Kết nối đến server khác: CONNECT <target server>[<port>[<remote server>]]

- Tìm đường đi tới server nào đó: TRACE [<server>]

- Tìm người quản lý (admin) của server: ADMIN [<server>]

- Yêu cầu thông tin từ server: INFO [<server>]

### 3.4. Nhóm message gửi text

- Private message: dùng cho client, user có thể gửi private message từ user đên user nhận, và chỉ có user nhận mới nhận được private message, cú pháp: PRIVMSG <receiver> {,<receiver} <text to be sent>

- Notice message: cũng tương tự như private message, nhưng khác nhau là không cần có sự đáp lại từ user nhận, cú pháp: NOTICE <nickname> <text>

### 3.5. Nhóm message do client truy vấn đến server (user-based query)

- Who query: là message dùng cho client để tạo ra truy vấn, kết quả truy vấn là danh sách các user phù hợp với đối số <name>, cú pháp: WHO [<name>[<o>]]

- Who is query: là message dùng yêu cầu thông tin cụ thể về một user nào đó, cú pháp: WHOIS [<server>] <nickname> [,<nickmask>[,…]]

- Whowas message: là message yêu cầu thông tin những user trong quá khứ mà hiện không còn có mặt trong danh sách user active, cú pháp: WHOWAS <nickname> [<count[<server]]

### 3.6. Nhóm message khác (miscellaneous message)

- Kill message: được dùng khi cần kết thúc kết nối giữa client và server, cú pháp: KILL <nickname> <comment>

- Ping message: dùng để kiểm tra một client còn hoạt động trong hệ thống mạng này không, nếu sau time out mà không trả lời thì client coi như đã bị ngắt kết nối, vì vậy khi nhận được Ping message thì client cần trả lời Pong message càng sớm càng tốt, cú pháp: PING <server l> [<server2>]

- Pong message: là message mà client gửi lại server để xác nhận chưa bị ngắt kết nối, cú pháp: PONG <daemon> [<deamon2>]

- Error message: dùng cho server thông báo lỗi cho operator, cú pháp: ERROR <error message>

### 3.7. Nhóm message tùy chọn (option message)

- Away message: là loại message mà client thiết lập để trả lời tự động cho private message, cú pháp: AWAY [message]

- Rehast message: là lệnh do operator thi hành bắt buộc server phải đọc lại và xử lý tập tin cấu hình của server đó, cú pháp: REHASH

- Restart message: là message dùng cho channel operator buộc server phải khởi động lại hệ thống, cú pháp: RESTART

- Summon message: mời một client có chương trình server hiện đã có cài đặt trên host gia nhập hệ thống IRC, cú pháp: SUMMON <user> [<server>]

- User message: dùng để trả về danh sách của các user đã login vào server, tương tự như WHO message, cú pháp: USER [<server>]

- Operwall message: khi wallops hoạt động, nó sẽ gửi <text..> dến tất cả các operator hiện có trên mạng, cú pháp: WALLOPS <text to be send>

- Userhost message: là message yêu cầu trả về thông tin của client có nickname làm đối số, cú pháp: USERHOST <nickname> {<space><nickname>}

- ISON message: là message được thiết lập để trả về thông tin về <nickname> hiện thời trên IRC một cách hiệu quả nhất, cú pháp:ISON <nickname> {<space> <nickname>}

# **CHƯƠNG 2: MÔ TẢ GIAO THỨC INTERNET RELAY CHAT**

## **1. Mô hình Client kết nối đến Server**

**Client**

**Server**

No

No

No

Yes

Yes

Yes

No

No

Yes

No

Yes

Yes

Start

Listen on port

Receive

Detect error?

Receive

Detect error?

Receive

Detect error?

Detect error?

Receive

Detect error?

Receive

Time out?

Start

Receive

000000000000000000000000000000

## **2. Mô hình client tham gia vào channel**

**Server**

**Client**

Receive

Yes

Detect error?

Time out?

No

No

Yes

Receive

Receive

## **3. Mô hình thoát khỏi channel và hủy bỏ kết nối**

Receive

END

Receive

No

Yes

Yes

No

Detect error?

Receive

**Server**

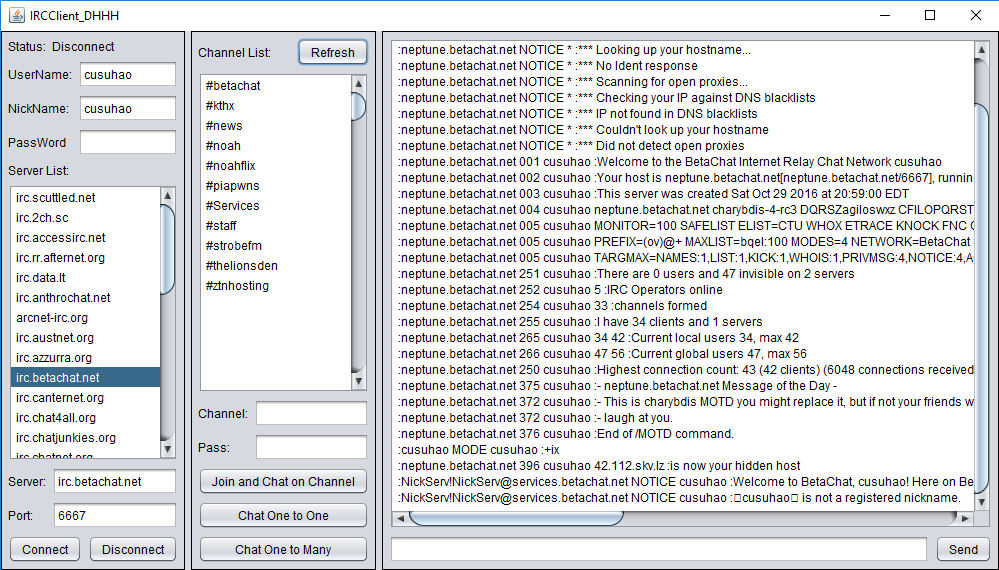
Time out?

**Client**

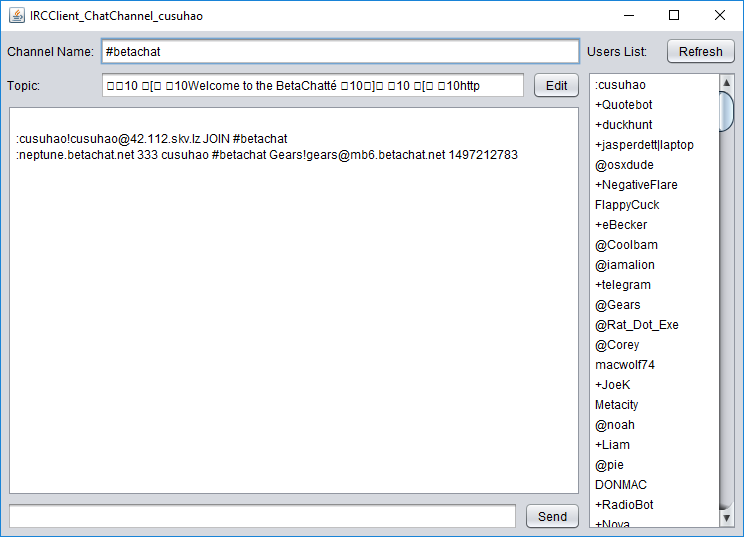
# **CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH**

## **1. Thiết kế chương trình**

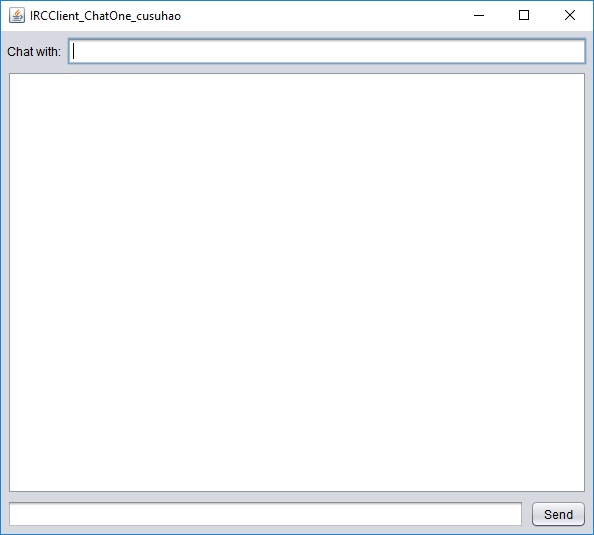
### 1.1. Giao diện của chương trình



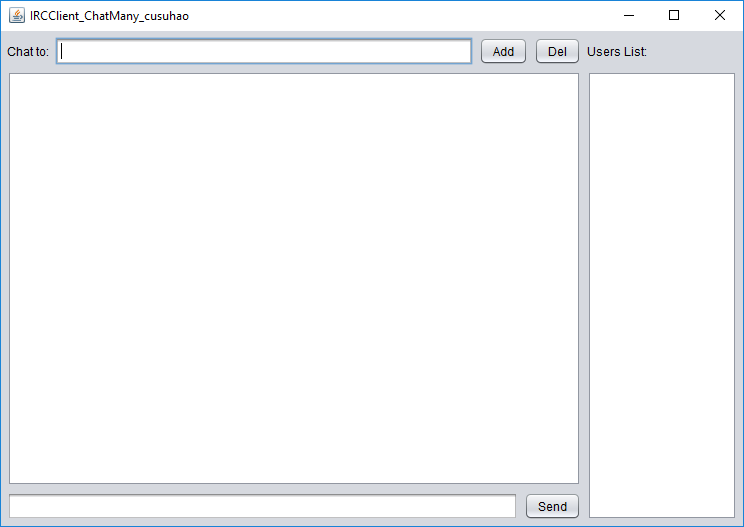
Hình 3.1: Giao diện khởi động của chương trình IRC



Hình 3.2: Giao diện cửa sổ Chat on Channel



Hình 3.3: Giao diện cửa sổ Chat One to One



Hình 3.4: Giao diện cửa sổ Chat One to Many

### 1.2. Danh sách chức năng

- Chức năng kết nối và ngắt kết nối client tới server

- Chức năng gửi message tới server

- Chức năng nhận message từ server

- Chức năng xử lý message nhận được từ server để đưa ra hiển thị thích hợp như hiển thị danh dánh channel, hiển thị danh sách user trên channel, hiển thị topic của channel, đưa message chat tới đúng cửa sổ cần hiển thị, …

- Chức năng tham gia channel, chat trong channel, sửa topic của channel, hiển thị danh sách user trong channel

- Chức năng chat với một người cụ thể nào đó

- Chức năng gửi message cho một số người cụ thể

## **2. Cấu trúc của chương trình**

Chương trình gồm hai package là Interface và Process

### 2.1. Package Interface

- Gồm các class dùng để hiển thị giao diện cho người dùng

a) Class MainGUI: hiển thị giao diện chính của chương trình

- Thuộc tính: các JLabel, các JBotton (btConnect, btDisconnect, btRefresh,…), các JTextField (textUser, textNick, textPass,…), JTextArea (textArea), JList (listServer, listChannel), ProcessIO (psIO), PsMain (psMain)

- Phương thức: btConnectActionPerformed (kết nối client tới server), btDisconnectActionPerformed (ngắt kết nối client tới server), btRefreshActionPerformed (gửi message LIST tới server để làm mới listchannel), btChatChannelActionPerformed (JOIN vào chanel và mở cửa sổ ChatChannel), btChatOneActionPerformed (mở cửa sổ ChatOne để chat với một người),  
btChatManyActionPerformed (mở cửa sổ ChatMany để chat tới nhiều người), btSendActionPerformed (gửi message có trong textChat đến server), listServerMouseClicked (setText textServer là name trong listserver đã chọn), listChannelMouseClicked (setText textChannel là name trong list đã chọn), formWindowClosed (thực hiện ngắt kết nối tới server trước khi cửa sổ chính bị đóng).

b) Class ChatChannelGUI: hiển thị giao diện Chat Channel

- Thuộc tính: các JLabel, các JButton (btEdit, btRefresh, btSend), các JTextField (textChannel, textChat, textTopic), JTextArea (textArea), ProcessIO (psIO), PsChatChannel (psCC), int nPsCC.

- Phương thức: btEditActionPerformed (gửi message edit TOPIC tới server), btRefreshActionPerformed (gửi message Topic và Name tới server để nhận topic và danh sách user có trong channel để làm mới danh sách user), btSendActionPerformed (gửi message có trong textChat tới channel), formWindowClosed (thực hiện Part Channel trước khi cửa sổ bị đóng, giải phóng bộ nhớ các biến có liên quan tới xử lý cửa sổ).

c) Class ChatOneGUI: hiển thị giao diện Chat One to One

- Thuộc tính: JLabel, JButton (btSend), JTextField (textChat, textUser), JTextArea (textArea), ProcessIO (psIO), PsChatOne(psCO), int nPsCO.

- Phương thức: btSendActionPerformed (gửi message có trong textChat tới người cụ thể có trong textUser), formWindowClosed (giải phóng bộ nhớ các biến có liên quan tới xử lý cửa sổ)

d) Class ChatManyGUI: hiển thị giao diện Chat One to Many

- Thuộc tính: các JLabel, các JButton (btAdd, btDel, btSend), JTextField (textChat, textUser), JTextArea (textArea), JList (listUser), ProcessIO (psIO), String (listUsers[], users), int nUser.

- Phương thức: btAddActionPerformed (thêm user trong textUser vào listUser), btDelActionPerformed (xóa user có tên trong textUser ở trong listUser), btSendActionPerformed (gửi message có trong textChat tới danh sách listUser), listUserMouseClicked (setText textUser là user đã chọn trong listUser).

### 2.2. Package Process

a) Class ProcessIO: thực hiện kết nối, ngắt kết nối client tới server, gửi message, nhận message, truyền message nhận được tới các class xử lý message.

- Thuộc tính: String(user), Socket (socketOfClient), BufferedWriter (writer), BufferedReader (reader), ReceiverThread (receive), PsMain (psMain), PsChatOne (psCO[]), PsChatChannel (psCC[]), int (nPsCO, nPsCC).

- Phương thức: addCO (thêm class xử lý giao diện ChatOne), delCO (xóa class xử lý giao diện ChatOne), addCC (thêm class xử lý giao diện ChatChannel), delCC (xóa class xử lý giao diện ChatChannel), connect (kết nối client tới server), disconnect (ngắt kết nối client tới server), send (gửi message tới server), Class ReceiveThread (tạo luồng nhận message từ server), psMessage (truyền message nhận được đến các class xử lý cụ thể.

b) Class PsMain:

- Thuộc tính: JList (listServer), JList (listChannel), JTextArea (textArea), boolean (getListChannel), String (listChannels[]), int nChannel;

- Phương thức: getListChannel (kiểm tra message nhận được có phải message chứa thông tin về danh sách channel ko, nếu có thì hiển thị danh sách đó lên listChannel), showMessage (thực hiện hiển thị message nhận được lên textArea)

c) Class PsChatChannel:

- Thuộc tính: JTextField (textTopic, textChannel), JTextArea (textArea), JList (listUser), String (listUsers[], topic).

- Phương thức: psMessage (thực hiện chuyển message nhận được tới các hàm kiểm tra message trong class), checkJoin (kiểm tra message nhận được có chứa thông tin về hành động Join hoặc Part channel không, nếu có thì hiển thị message đó lên textArea), getTopic (kiểm tra message nhận được có chứa thông tin topic của channel không, nếu có thì hiển thị topic của channel lên textTopic), getListUser (kiểm tra message nhận được có chứa thông tin về danh sách user của channel không, nếu có thì đưa danh sách đó lên listUser), getChat (kiểm tra xem message đó là message chat gửi đến channel không, nếu có thì hiển thị message lên textArea).

d) Class PsChatOne

- Thuộc tính: String user, JTextArea textArea, JTextField textUser

- Phương thức: psMessage (kiểm tra message là message chat gửi đến user đang chat trong cửa sổ này không, nếu có thì hiển thị lên textArea).

## **3. Hoạt động của chương trình**

### 3.1. Hoạt động kết nối

B1: Thực hiện gọi hàm connect của đối tượng psIO và truyền vào User, Nick, Pass, Server, Port khi click vào button Connect trên giao diện chính.

B2: Hàm connect sẽ tạo socket theo Server và Port được truyền vào, sau đó khởi tạo biến để gửi, nhận dữ liệu.

B3: Hàm connect thực hiện khởi tạo và bắt đầu luồng nhận dữ liệu từ server.

B4: Hàm connect thực hiện gửi Pass, Nick, User đến server.

B4: Nếu connect thành công, hàm trả về true, và trả về false nếu ngược lại.

### 3.2. Hoạt động ngắt kết nối

B1: Thực hiện gọi hàm disconnect của đối tượng psIO khi click vào button Disconnect trên giao diện chính.

B2: Hàm disconnect thực hiện gửi message “QUIT :Goodbye\r\n” tới server và thực hiện đóng luồng gửi, nhận dữ liệu.

B3: Nếu disconnect thành công, hàm trả về true, và false nếu ngược lại.

### 3.3. Hoạt động Open Chat on Channel

B1: Thực hiện gửi message Join vào channel đến server, sau đó khai báo và khởi tạo đối tượng ChatChannelGUI, đồng thời truyền vào tên của channel, psIO khi click vào button Join and Chat on Channel trên giao diện chính.

B2: Đối tượng ChatChannelGUI sẽ khai báo và khởi tạo đối tượng PsChatChannel và truyền vào textChannel, PsChatChannel xem textChannel như ID của cửa sổ ChatChannelGUI để xử lý các message đến để có thể truyền đến đúng cửa sổ.

B3: Đối tượng ChatChannelGUI gửi các message TOPIC và NAMES đến server

B4: Đối tượng PsChatChannel sẽ xử lý các message để nhận topic và users của channel để hiển thị lên ChatChannelGUI tương ứng.

B5: Khi thực hiện đóng cửa sổ, đối tượng cửa sổ sẽ thực hiện gọi đến hàm delCC trước. Hàm delCC nhận tham số vào là vị trí đối tượng xử lý của sổ này, nó thực hiện xóa đi đối tượng đó.

### 3.4. Hoạt động Open Chat One

B1: Thực hiện khai báo và khởi tạo đối tượng ChatOneGUI, đồng thời truyền vào username, psIO khi click vào button Chat One to One trên giao diện chính.

B2: Đối tượng ChatOneGUI sẽ khai báo và khởi tạo đối tượng PsChatOne và truyền vào textUser, PsChatOne xem textUser như ID của cửa sổ ChatOneGUI để xử lý các message đến có thể truyền đến đúng cửa sổ.

B3: Khi thực hiện đóng cửa sổ, đối tượng cửa sổ sẽ thực hiện gọi đến hàm delCO trước. Hàm delCO nhận tham số vào là vị trí đối tượng xử lý của sổ này, nó thực hiện xóa đi đối tượng đó.

### 3.5. Hoạt động của quá trình nhận message

B1: Luồng nhận dữ liệu được khởi tạo và được bắt đầu sau khi client kết nối tới server

B2: Khi nhận được message, luồng gọi hàm psMessage trong class ProcessIO và truyền vào message.

B3: Hàm psMessage sẽ gọi đến các hàm psMessage trong các đối tượng PsChatOne, PsChatChannel, PsMain hiện có.

B4: Tại class PsChatOne, hàm psMessage sẽ kiểm tra message và hiển thị lên textArea nếu người gửi đúng với textUser (được coi là ID của cửa sổ).

B5: Tại class PsChatChannel, hàm psMessage sẽ gọi đến hàm checkJoin để kiểm tra xem message chứa thông tin các user join vào kênh không, nếu có thì hiển thị lên textArea, nếu không sẽ gọi đến hàm getTopic để kiểm tra xem message có chứa thông tin topic của channel không, nếu có thì hiển thị topic channel lên, nếu không thì gọi đến hàm getListUser để kiểm tra xem message có chứa thông tin danh sách user của channel không, nếu có thì cập nhật danh sách user, nếu không thì gọi hàm getChat để kiểm tra đây có phải message chat gửi đến channel không, nếu có thì message sẽ hiển thị lên textArea, nếu không thì hàm psMessage trả về false.

B6: Tại class psMain, hàm getListChannel sẽ được gọi đến để kiểm tra xem message chứa thông tin danh sách channel không, nếu có thì cập nhật và hiển thị danh sách channel, nếu không thì hàm showMessage được gọi để hiển thị message lên textArea của giao diện chính.

# **CHƯƠNG 4: THỬ NGHIỆM VÀ ĐÁNH GIÁ CHƯƠNG TRÌNH**

## **1. Kiểm thử chương trình**

### 1.1. Yêu cầu của chương trình

- Môi trường hoạt động: Java 8 update 131

- IDE sử dụng: NetBeans IDE 8.2

### 1.2. Kịch bản kiểm thử

B1: Khởi động chương trình

B2: Kết nối client đến server

B3: Gửi message từ client đến server trên giao diện chính

B4: Chọn channel trong danh sách channel hoặc nhập tên channel khác và JOIN

B5: Sửa Topic trong channel

B6: Gửi message trên giao diện channel

B7: JOIN vào 2 channel khác, thực hiện các bước tương tự

B8: Thoát channel đầu tiên, thực hiện gửi message ở các channel còn lại

B9: Khởi động thêm 2 (hoặc 3, 4) chương trình IRC Client

B10: Kết nối 2 client này đến cùng server client đầu tiên đã kết nối

B11: Thực hiện cho 3 client Join vào cùng 1 channel và gửi message trên channel

B12: Mở Chat One to One trên mỗi client và thử gửi message qua lại giữa các client

B13: Đóng 1 trong các cửa sổ Chat One to One trên và tiếp tục gửi message

B14: Mở Chat One to Many tại client đầu tiên, nhập user nhận là 2 client còn lại, gửi message đến 2 client còn lại.

B15: Ngắt kết nối 1 trong các client tới server

B16: Gửi message từ client còn hoạt động đến client đã ngắt kết nối

B17: Kết thúc chương trình và kết thúc kiểm thử

## **2. Đánh giá chương trình**

### 2.1. Ưu điểm

- Chương trình hoạt động ổn định.

- Các chức năng đã có hoạt động ổn định.

- Giao diện thiết kế dễ dùng, dễ thao tác.

- Các cửa sổ con bật lên không giới hạn, tùy thuộc vào bộ nhớ RAM máy tính sở hữu.

### 2.2. Nhược điểm

- Chương trình không kết nối được nhiều server, muốn kết nối đến server khác, cần khởi động thêm một chương trình khác.

- Chức năng hiển thị topic của channel chưa hoàn thiện, khi channel có topic quá dài, topic không được hiển thị hết lên textTopic

- Chức năng JOIN channel chưa hoàn thiện, khi thực hiện click vào button Join and Chat on Channel mà không thể Join vào channel thì cửa sổ ChatChannel vẫn hiện ra, chỉ hiển thị message Cannot Join Channel lên textArea của cửa sổ mà không tự động đóng cửa sổ đó.

- Khi thực hiện ngắt kết nối đến server, các cửa sổ ChatChannel, ChatOne, ChatMany hiện có không tự động đóng, mà phải thao tác trực tiếp.

- Chương trình chỉ hỗ trợ chat mà không hỗ trợ chia sẻ tập tin và tài liệu.

# **KẾT LUẬN**

IRC là một mạng trò truyện trực tuyến lớn, tốc độ liên lạc giữa client với server và giữa các client rất cao. Chương trình IRC client cho phép người dùng có thể giao tiếp, gặp gỡ nhau mà không phải gặp trực tiếp. Chương trình hỗ trợ trao đổi thông tin dưới nhiều hình thức khác nhau như chat riêng giữa hai người cụ thể, một nhóm người chat với nhau, hoặc một người có thể gửi tin nhắn tới nhiều người theo một danh sách cụ thể. Một số chương trình IRC client phổ biến còn cho phép chia sẻ tập tin và tài liệu cho nhau và hỗ trợ chạy trên nhiều hệ điều hành khác nhau.

Qua thời gian tìm hiểu và xây dựng chương trình, em thấy chương trình mà mình đã xây dựng còn rất nhiều hạn chế, và nhược điểm. Nếu có thời gian và điều kiện tiếp tục phát triển đề tài, em sẽ hoàn thiện thêm các chức năng cho chương trình cũng như thiết kế giao diện chương trình thân thiện, dễ dùng hơn để chương trình hoàn thiện hơn.