CS494 Julian Saunders
IRC Project Specification Portland State University
5/16/18

Abstract

This memo describes the communication protocol for an IRC-style client/server system for the Internetworking Protocols class at Portland State University.

Table of Contents

## 1. Introduction

This specification is for an Internet Relay Chat (IRC) protocol that will enable multiple client programs to connect to a server program. Any messages sent to the server from a client will be sent to every connected client.

Users of the client program will be able to create a new chat room on the server, join an existing chat room, leave a chat room, and display all currently existing chat rooms. Messages sent in a chat room will only be received by other user clients that are currently connected to that specific room.

## 2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

In this document, these words will appear with that interpretation only when in ALL CAPS. Lower case uses of these words are not to be interpreted as carrying significance described in RFC 2119.

## 3. Basic Information

The client and server programs communicate via TCP/IP protocols using a port designated by the user in the UI. The client will send information to the server once the user has typed at least one character and hit enter. The server will then broadcast the message to all other users in the chat server. Each chat server is setup on a seperate port. To terminate a chat server correctly a message must be sent from a client to the server.

The user will be greeted with a user interface upon starting up the client program. This interface will give the user various options for interacting with server and chat rooms. The user may issue a command to leave a chat room and return the the user interface.

## 4. User Interface Functions

### 4.1. Start Client

run_client(host,Port)

#### 4.1.1. Usage

Calling the client function will launch the client with the corresponding hostname and port that are passed to it.

Client MUST initiate a connection to the server before the user can issue any commands. If a connection cannot be made, the handshake SHOULD return an error informing the user before the client program is terminated.

4.1.2. Field Definitions

s - socket connection python call
socket_list - variable list to hold currently connected sockets
read/write/error _sockets - python library values used by the select function
sock - Incoming message over read socket
data - buffer to hold incoming message from server
msg - User input from stdin
stringmsg - Variable to hold decoded response to check for opcodes

4.2. List Rooms

print_ports()

4.2.1. Usage

User interface reads file that contains the list of current ports for active chat servers.

4.2.2. Response

UI MUST return a list of every existing room that is available to join.

4.3. Create and Join a Room

run_server (port) [Threaded]
run_client(host,port)

4.3.1. Usage

User interface starts a server on a seperate thread followed by a call to start the client. Both server and client will run on the designated port number and default host.

4.3.2. Field Definitions

t1 - variable holding the server thread
threads - variable holding a list of all currently active server threads

## 5. Client / Server Messages

### 5.1. Sending Messages to Clients

send_data (sock, message, client_list, server_socket)

#### 5.1.1. Usage

Server should take messages meant for broadcast and send them to the clients that are associated with the chat room the message originated from.

#### 5.1.2. Field Definitions

sock - current client socket
message - message to be sent to client
client_list - list of clients used by calling function
server_socket - server socket used by calling function

### 5.2 Leaving a Room

While in the chat client, user may type “client exit” (without quotes) to return to the UI. If the user was hosting a server, the server will continue to run and be joinable.

### 5.3 Shutting Down a Server

From a chat client, the user may type “server exit” (without quotes) to properly shutdown a server. This will exit the client as well as the user interface and any other servers the user was hosting.

## 6. Error Handling

If the client loses the server connection then the user must be notified that the server connection was lost before the client is terminated. If the server loses connection to a client, it may send a notification to the console for debug.

## 7. Security Considerations

This protocol has no encryption so any information sent from the client or the server can be seen on the network.