A Simple Peer-to-Peer Social Network

NETWORK Machine Problem 1

2nd Term AY 2014-2015

1 Introduction

You will be creating a peer-to-peer social network application. This machine problem will be done in pairs. This is a sockets programming exercise and hence graphical user interface is not given credit until all the networking requirements are fulfilled. A program that accepts input from the user and displays posts via the command prompt may suffice to get a 100% grade. Any programming language is allowed. It is assumed that an IP address is associated to only one user, hence no login and logout is required.

2 Social Network Protocol

Each peer must be able to send and receive the following commands through network sockets with their appropriate syntax. In the case of multiple syntax the receiving peer must be able to understand all of them, though not necessarily support all of them. For example, in the FOLLOW command below, a peer that does not support profile-pictures must still be able to deduce the FOLLOW message it receives in any syntax. Any message that does not follow these formats should be discarded.

Note that these messages are the ones sent through the network and are not necessarily what is typed in the command prompt. You will have to decide what are the command prompt input arguments needed.

All usernames must be less than or equal to 15 characters excluding the NULL character. As shown below there are two usernames the *follower* and *followee*. The *follower* is the peer that requests to receive broadcast messages from *followee*

Images are in JPEG format and are streamed in the network in binary format. The images mentioned in this document are profile pictures (*profile-picture*) and images in posts (*img*). Profile Pictures are images that the user wishes to appear whenever he sends a message. Note that we use the term *followers* for peers that wants to receive our posts and *followees* for peers that we are receiving posts from.

2.1 Follow Request (FOLLOW)

Syntax:

"FOLLOW" NULL

"FOLLOW" SP follower NULL

"FOLLOW" SP follower NULL profile-picture EOF

This command is sent if a certain user wishes to follow another user. The request can be sent with the parameter *follower*, which is the username of the follower.

2.2 Follow Request Approved (APPROVE)

Syntax:

"APPROVE" SP followee NULL

"APPROVE" SP followee profile-picture EOF

An APPROVE message is sent by a peer that receives a follow request message to tell the follower that the latter's follow request has been granted.

2.3 Unfollow Request (UNFOLLOW)

Syntax:

"UNFOLLOW" An unfollow request is sent by a follower to tell its peer that it no longer wants to receive posts from that peer.

2.4 Messaging Broadcasting (POST)

Syntax:

"POST" SP message NULL

"POST" SP message EOF img EOF

A peer sends messages to his followers by sending a POST message. Messages must be in ASCII format and should end with a NULL character. The message length should not exceed 256 characters excluding the NULL character. Another version of posting allows the user to not only send a message but also send an image along with it.

2.5 Private Messaging (PM)

Syntax:

"PM" SP message NULL

"PM" SP message EOF img EOF

A peer that wishes to send a private message must send a PM command to the user.

2.6 Profile Picture Update (IMG)

Syntax:

"IMG" SP profile-picture EOF

An IMG message signifies the intent of a peer to update his profile picture. Peers must properly associate profile pictures to their respective users.

2.7 File Transfer (FILE)

Syntax:

"FILE" SP filename SP file EOF

Aside from images, other file types can also be sent by their peers to their followers. The *file* parameter is the binary data content of the file with the parameter *filename* referring to the name of the file. Files must not exceed 65500 bytes.

3 Checking

Checking of the machine problem will be done in class. The whole class would follow and unfollow each other and send and receive posts from one another. Each group must be able to connect with other groups in the network.

Milestone 1: 50%

The program can both send and receive messages at the same time. *What programming concept must you use in order to do this?* Note that the IP address is not included in the network message sent but it will be needed as a command line argument. *Why is an IP address needed in the command line but not in the network message?*

Milestone 2: 60%

The program can determine the IP address of the message sender. *What*

sockets programming function helps you do this? How does it work?

Milestone 3: 70%

The program can follow peers and accept followers. There must also be a mechanism in which the username is associated with the IP address of the peer such that when a FOLLOW request is received, the receiver can know in which IP address to reply to. The program must also process the UNFOLLOW command. *Explain how you implemented this.*

Milestone 4: 75%

The program sends posts to followers.*How do you make sure that you are only sending posts to your followers? What is the maximum number of characters you can send in the socket?* Implement an error-checking mechanism that make sure that the message does not exceed this limit.

Milestone 5: 80%

The program displays posts from followees. The username of the sender must be displayed. *How do you know the username of your followees even when the POST message does not have an IP address or a username?*

Milestone 6: 85%

The program can accept private message. The username of the sender must be displayed and indicate that it is a private message.

Milestone 7: 90%

The program sends and receives profile-pictures. There is no need to display yet, just store it in a folder. Make sure that the profile pictures are associated with the proper user. *What data type is needed to implement this?*

Milestone 8: 100%

The program sends files to followers and receives files from followees.

Milestone 9: 110%

Has a graphical user interface. The profile-pictures should be displayed in the graphical user-interface.

Milestone 10: 120%

(JPG, GIF, BMP) sent through posts and private messages should be displayed. Audio files (MP3, WAV) should also be playable.

4 Submission

A final documentation which should be submitted before midnight of October 13 and 14 for the MW and TH class respectively in PDF format. Your final documentation must explain how each milestone is reached and answer the questions accordingly. All machine problem codes and the final documentation should be sent to your instructor's email address with a subject of "NETWORK [section] Final Docu". Make sure your codes are compiles without any error.

Email Address:
mac.cordel@delasalle.ph
ann.laguna@dlsu.edu.ph