**Operating Systems**

**CS4348**

**Project #3: Network Communication Using Sockets**

**Due Date: Saturday, December 5, 2015**

## I. Project Organization

This project will utilize Sockets for communication between processes.

You should do the following pieces to complete your project. Each piece is explained below:

* Code 50 points
* Output 40 points
* Summary 10 points

# Code

Your code should be nicely formatted with plenty of comments. The code should be easy to read, properly indented, employ good naming standards, good structure, etc.

# Output

Output will be graded by running your program.

# Summary

The summary section should discuss your project experience. You should discuss how you approached the project, including anything difficult or interesting, what was learned, and the end result. This should be at least 1 page in length.

**Teams**

This project may be performed alone or in a team of 2 people. In the case of a team, the work should be performed collaboratively with equal participation or divided evenly. The summary should include discussion of what each team member did. Each team member should be prepared to explain the code if asked to do so.

## II. Project Description

**Language/Platform**

This project must target a Unix platform and execute properly on our CS1 server.

The project must be written in C, C++, or Java.

If using C or C++, you must use the Sockets API with stream sockets.

If using Java, you must use java.net.Socket and java.net.ServerSocket.

Your approach should be similar to the examples given in class.

For C/C++, you must only use read and write system calls to communicate over the socket.

For C/C++, you must account for possible partial messages as discussed in class. This means using a loop to read or write all expected bytes. You should not make a separate call for each byte.

For Java, you may use any mechanism Java provides for synchronization and mutual exclusion.

Any other approaches require instructor approval.

### Client/Server

This project will utilize Sockets for communication between processes.

Your task is to write a simple client and server.

The server will maintain a list of simulated files and allow the clients to perform operations on them.

The server will:

1. Accept a port number as a command line argument.
2. Accept connections from clients.
3. Create a simulated list of files, each one having a name and associated contents, which can be plain text or an image. The server can read these in initially, but will not interact with the real file system thereafter.
4. Create a new thread for each client.
5. End by termination with control-C.

The server thread will:

1. Accept and process requests from the client.
2. Provide mutual exclusion protection for the simulated file system.
3. Send only the minimal data needed to the client, not the menu or other UI text.

The client will:

1. Accept a machine name and port number to connect to as command line arguments.
2. Connect to the server.
3. Prompt for a command from the user.
4. Present the following menu of choices to the user:
   1. Display the names of all files.
   2. Get a particular file. For a command-line interface, save the file to the local directory and inform the user it was saved. (If using a GUI, can display either type of file without saving it).
   3. Remove a file from the list.
   4. Add a new file to the server.
   5. Exit.
5. Interact with the server to support the menu choices.
6. Ask the user for the next choice or exit.

## III. Project Guidelines

### Submitting

Submit your project on eLearning. Include in your submission the following files:

1. readme.txt. A readme file describing how to compile and run your project
2. team\_members.txt list of team members, only required for teams
3. summary.doc A Word document for the summary
4. Your source files

For a team submission, only one person should submit.

### Academic Honesty

All work must be your own. If cheating is suspected, you will be referred to the Judicial Affairs Office for further discussion. Copying may be detected in a number of ways, including by software which compares your code with all other students’ source code, by comparison with code on the Internet, or by a visual inspection of your source code.

### Resources

The web has many articles on Sockets. There are also books available on Sockets. The course website also contains an example of Socket source code.