**Operating Systems**

**CS4348**

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

**Due Date: Saturday, December 7, 2013**

## 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:

* Design 15 points
* Code 50 points
* Output 25 points
* Summary 10 points

# Design

The messages exchanged between client and server should be documented in a table, listing each message sent, the purpose of the message, and a description of the data it contains.

# 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 or divided evenly. The summary should include discussion of what each team member did. Each team member should understand all of the submitted work and be able to answer questions about it.

## II. Project Description

**Language/Platform**

This project must target a Unix platform and execute properly on our Unix netxx or CS1/CS2 servers.

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. It is not permitted to process one byte at a time for a message.

### Client/Server Demonstration

This project will utilize Sockets for communication between processes.

Your task is to write a client and a server.

The server will simulate a Mars rover. The client will send commands to the rover to perform.

The server will:

1. Accept a port number as a command line argument.
2. Accept connection from a client.
3. Read a command from the client.
4. Perform the command and send response back to client.
5. Process client requests until the client exits.
6. Send only the raw 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. Present a menu of choices to the user.
4. The user may:
   1. Turn left 90 degrees.
   2. Turn right 90 degrees.
   3. Take a picture.
   4. Get direction, initially facing north.
   5. Get the air temperature.
   6. Exit.
5. Interact with the server to support the user queries.
6. Ask the user for their next choice or exit.

Miscellaneous:

1. The air temperature is a random number from -50 C to 50 C.
2. The picture will be one of four files of the landscape of Mars that you provide.
3. The four files correspond to the four directions the rover can face (North, West, South, East).
4. Turning allows the corresponding photo for the new direction to be sent.
5. The interface may be embellished beyond a menu-driven command-line interface.

Sample output:

{cslinux1:~/sample} server 2024

Mars rover running on CS1 port 2024

Client connected.

Rover sends image.

Rover turns left 90 degrees.

Rover sends air temperature of -10 C.

Client disconnected.

{cslinux1:~/sample} client CS1 2024

Client connected to rover on CS1 port 2024

Choose from the following list of commands:

1. Turn left 90 degrees.
2. Turn right 90 degrees.
3. Take a picture.
4. Get direction.
5. Get the air temperature.
6. Exit.

Enter choice: C

Image1.jpg saved.

Choose from the following list of commands:

1. Turn left 90 degrees.
2. Turn right 90 degrees.
3. Take a picture.
4. Get direction.
5. Get the air temperature.
6. Exit.

Enter choice: A

Rover has turned left 90 degrees.

Choose from the following list of commands:

1. Turn left 90 degrees.
2. Turn right 90 degrees.
3. Take a picture.
4. Get direction.
5. Get the air temperature.
6. Exit.

Enter choice: E

Air temperature at Mars rover is -10 C.

Choose from the following list of commands:

1. Turn left 90 degrees.
2. Turn right 90 degrees.
3. Take a picture.
4. Get direction.
5. Get the air temperature.
6. Exit.

Enter choice: F

Disconnected from Mars rover.

## 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 your project
2. design.doc. A Word document for the design
3. summary.doc A Word document for the summary
4. Your source files

### Academic Honesty

All work must be your own. If cheating is suspected, you will be referred to the Judicial Affairs Office for further discussion. Cheating may be detected by software which compares your code with all other students’ source code, 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.