Design Of Graphical User Interface And Server and Client System Of Networked Aquatic Microbial Observing System

Koto Norose
Undergraduate Researcher

Arvind Pereira
Graduate Student Mentor

Bin Zhang
Graduate Student Mentor

Amit Dhariwal
Graduate Student Mentor

Gaurav Sukhatme
Faculty Mentor

norose@eng.utah.edu, {menezesp, binz, dhariwal, gaurav}@usc.edu

Technical Report
Robotic Embedded Systems Lab
Department of Computer Science
University of Southern California

Aug. 16, 2006

ABSTRUCT

This paper describes the design of a simple Graphical User Interface (GUI) and server and client system for the Networked Aquatic Microbial Observing System (NAMOS). The network has ten static buoy nodes and one mobile robotic boat to measure the chemical, physical and biological phenomena. The goal of the network is to gain high-resolution information of plankton assemblages in aquatic environment and hydrographical requirements. This network will contribute to enable (1) real-time observation in aquatic ecosystems and (2) sensor actuated sampling for biological analysis. The objective of the GUI and server and client system for NAMOS is to enable a user to select a buoy, connect to it, get data of a specific time period, and plot these data without knowing the command line.

1. INTRODUCTION

This report describes the design of a simple Graphical User Interface (GUI) and server and client system for the Networked Aquatic Microbial Observing System (NAMOS). NAMOS is a research project involving robotics, sensor networks and marine biology. The goal of NAMOS is to perform adaptive hydrographic sampling using a network consisting of buoys and boats, and to develop strong, decentralized algorithms and hardware to enable this task. The objective of the GUI and server and client system for NAMOS is to enable a user to select a buoy, connect to it, get data of a specific time period, and plot these data without knowing the command line.

2. SYSTEMS AND ALGORITHM TO GATHER DATA OF A SPECIFIED TIME RANGE

The buoy works as a server and the laptop works as a client. The client will send a request to the server and the server will send the requested data to the client. The fundamental server and client system has been developed using socket programming [1]. The server is written in C and the client is written in C#.

First, the date and time format was determined for the client GUI. The input style of the GUI will be the same style as the C# time stamp. Examples of this format are:

8/10/2006 4:13:45 PM
8/11/2006 0:0:3 AM

The advantage of this is that we can use built-in time stamp functions to easily convert between strings and numbers. The GUI will require two inputs: start time and end time. These will be sent to the server, and the server will send the data between these times.

Next, the client computer converts the input time to number of seconds since 1/1/1970 0:0:0 AM at the hardware on the buoy. Then the program calculates the difference between two inputs and sends starting time in seconds, the difference time in seconds, and the resolution in seconds to the server. The user can set the resolution by means of a track bar on the GUI. The resolution control enables the user to request and quickly receive high-resolution data over small range of time or low-resolution data over wide range of time.

Once the server receives the request from the client, the main thread of the server runs the algorithm to gather data for the specified time range. That is, first it finds the start time, then it finds the end time, then it calculates the number of samples to pick from the resolution, then it gathers and sends the data.

It first finds the start time. Since the server is written in C and the data are stored in a text file, the program searches from the end of the file and goes backwards, checking at each character for a new line. Until the new line has been reached, the program stores the characters in an array. If it finds a new line, then it reads the beginning of the stored array that contains the time in seconds. The program compares this time and the start time input, and stops if the found time is close enough to the start time input.

The program then goes back to the older data while the difference between the start time and the old data time is smaller than the requested input difference time using same method as used to find the start time.

After finding the proper time, the program picks the number of data corresponding the input resolution and stores them in an array. If the array isn't big enough, then the program sends an error message, otherwise it sends the array.

When the client gets the data from server, and if the data is not the error message, then the client saves the data to a text file. Consequently, the client calls the plotting software to plot the data and save that plot as a picture file. Finally, the client shows that picture file to the user.

3. OTHER FEATURES

Several other features were added to the GUI. These are: One click connection and disconnection, the guide message for the user when the user set the cursor at the input, and status messages for the user. The status messages indicate whether the connection is ok or not, the scale of x-axis of the graph, etc.

4. TEST RESULT

The GUI was first simulated using a Linux machine and a data file from a past field trip. After this simulated successfully, real-time experiments were performed with the GUI and with the server executable running on the buoy. These experiments were also successful, although sometimes the data transfer took time and the plot picture didn't upload every time the user requested the data plot.

5. CONCLUSION

The GUI successfully connected to the buoy, got data from a specific time period, and plotted its graphs as expected. The GUI and server were validated using the hardware that will be used in actual field experiments, and the results were successful. Therefore, the GUI and server and client system are successfully developed. In the future, we could improve our GUI and server and client system as an oscilloscope in the Electrical Engineering but it is not limited.

REFERENCE

[1] A. Dhariwal, B. Zhang, C. Oberg, B. Stauffer, A. Requicha, D. Caron, and G. S. Sukhatme, "Networked Aquatic Microbial Observing System," in *IEEE International*

Conference on Robotics and Automation, Orland, FL, May 2006.