

Online Auctioneer

Joseph Cordina

May 2, 2009

This is the description for assignment 2 of unit CSA 2205, System Programming, for the year 2008/2009. This assignment is worth 13 marks of the total mark of this unit (i.e. 20% of the total marks for the Unix oriented systems programming part of this unit, the first assignment was worth 20 marks of the total mark of this unit). The deadline for this project is 22nd May 2009. You may do this assignment in a group of maximum 2 people, yet you may do this assignment individually also. A single mark will be allocated to the whole group, so choose your partner carefully and make sure that the work gets evenly distributed. Under no circumstances should code be shared outside your group. You will submit your assignment making use of the ASS system¹. Those of you without login access, please contact me on joseph.cordina at um.edu.mt. Each of you might be assigned a ten minute slot in which you will be presenting the implementation of your assignment and during which your program will be executed. The students that will be called for this presentation will be allocated at random by the administration office. I will run your program on two different systems (of different architectures) and if that works, you will pass your assignment (with a healthy amount of marks given). So really functionality is more paramount for this assignment, rather than design. The reason for this is to alleviate your load in terms of getting optimal efficiency and due to the fact that I am hoping that you will have assignments of other units that make use of socket programming. Please be reminded that you cannot copy and plagiarize to ease your way to a final submission. While you may discuss ideas with others, do not steal. You can find more information at

<http://www.cs.um.edu.mt/cs/links&resources/plagarism.html>

Read this information well since after wards no excuses will save you. Anyone resorting to such methods will be considered as trying to cheat and will even risk the removal from the degree program.

1 Submission Contents

You will submit your C files together with any accompanying header files (C language has to be used obviously) separately. Please do not use any tar or zip file formats. Additionally include any libraries you might have used. Ideally use *ncurses* as an interaction library. The compilation can assume the presence of 'gcc' on which your system will be compiled. Include with your submission *makefiles* that can compile your system². You will need to provide two makefiles, one to generate the **client** and another for the **server**. I will try your code by building your server and running it on any architecture. Then I will run separate instances on Free-BSD, SunOs and Linux machines in any order. With your submission, you will provide a file called report.txt which will be a very **brief** description of how your system works and provides the required functionality. Make sure you mention any special techniques or tools you made use of and also any considerations you have made to improve efficiency. More importantly, give details

¹www.cs.um.edu.mt/~jcord/ass

²You can find plenty of tutorials on the net that teach you how to set one up

of the network protocol between the interacting processes. Please do not include code inside this file, since this is obviously not readable. Additionally, please note down any bugs your system might have. Listing bugs is common practice, and will benefit the marking process. This file should also contain references to resources you have made use of in your program. Finally, you will submit a file 'protocol.jpg' containing an image that describes in the most concise and clear manner possible your system's architecture and messages that are passed between the components.

Before submitting, please do clean up your code. Remove any dead code and useless comments. Every system calls output should be validated for errors and when appropriate, an error message reported to standard output. Also do place some comments to describe your structures in the code, at least to show that you understand how your own program works. Remember C is only unreadable if you make it unreadable. I cannot emphasize enough to be careful with your naming conventions and visual formatting.

2 Let the Auction begin

Well the idea is simple, build the foundations of *ebay*³!! I hope all of you know what that is, otherwise you really should browse a bit more. So you will build a server that will manage auction items. An auction item is defined by the following

- ID of item (Integer)
- Name of item (String)
- Description of item (String)
- Date of opening of auction item (Date and Time ex 2009/04/13 05:30 for 13th April 2009 at 5:30 am, 24 hour clock assumed)
- Date of closing of auction item (Date and Time as above)
- Reserve, which is the least amount that the top bidder has to pay for him or her to win the product (amount is a float, reserve will not be seen by clients)

The idea is simple, one runs their `client` by passing it first the IP address and then the port number to use (ex. `client 193.188.34.2 6009`). Note any number of clients might be run on same machine, even using the same user name. When someone connects their `client` to the server over the Internet, they will be asked for a user name to use. Then they will be presented with the following menu

1. View All Items
2. View One Item
3. Bid for Item
4. Quit

When one presses the number for viewing all items, the list of items is shown according to the definition above. The items are sorted by first listing those actions that are open, then those that are still to open and finally those that have closed. The Reserve is not shown unless the client is connecting from the same machine of the server.

When one presses the number to view one item, he or she will be asked for the ID of the item, and then it will be shown as above. Now for both these options, if the auction is open or closed, the latest bidding price will be shown. In addition if the client is connecting from the same machine as the server, the user name of the highest bidder will be shown and if the auction is closed, the winner of the item and the winning

³www.ebay.com

amount will be shown. If reserve has not been met for closed auction items, this will also be indicated.

Finally if the user presses the bid items menu item, he will be asked for the ID of the item. Once this is entered and verified, the current status, details and highest bid of the item will be shown. If the highest bid is of the viewing user, then this will be indicated. Any new bid amount by anyone will refresh the view for any user bidding on the item. The user may enter any amount and once the user presses the return button, that bid will be registered. The usual rules of bidding will apply. Once the bidding finishes, the user will be told if he has won the item or otherwise. When ten seconds remain for closing of the bid, a count down value will be shown on screen. You can assume that all users are running within the same time zone. If the user presses 'q' at any point, he will be returned to the main menu screen.

2.1 Adding Auction Items

If the client is connected to the server from the same machine, an extra menu item will be shown as follows

1. Register new item

If this menu item is chosen, the user will be asked to enter the details of a new auction item. The ID of the item will be automatically given by the system. One has to enter all details and all relevant validation has to be performed. Once all details are entered, the user will be shown a preview of the item details and given a choice to activate the item or otherwise.

The client that registers an item can be closed after this step and the item will still be visible from other clients.

3 What might matter

Most of the marks will be allocated to a functional system that allows multiple clients to make requests to the servers. Additionally the most accurate auction rules application is expected. I will run several concurrent auctions and bids and if all works well its almost full marks for you!! It goes without saying that if you find an application implementing something similar you cannot use it. Obviously having a system that is free of race conditions, deadlocks and still considerably efficient would be nice. For your own good, first design your system before starting the coding. It will result in a better system and a quicker turn around time. Instances of the program will have to communicate with other instances through an Internet connection under non-privileged users.

You can assume that the server will always be started first. The server will always be run by passing it the port on which to listen in the command line (ex `server 6009`). Also you can assume that if the server is restarted, all auctions will be lost and the system will start from a clean configuration again.

This system has to be implemented using BSD stream sockets running on the TCP/IP protocol. Both the server and the client processes must be fault tolerant. They can both be terminated with a CTRL-C or the kill command. Ensure that proper cleanup is performed in each case. Thus the server must terminate all its child processes (if any) and close all open sockets. The child must close all its connections properly before terminating. A client must be fault tolerant to broken connections, i.e. it will not crash when the connection is broken but will react accordingly. Ensure that you develop an appropriate protocol between the client and the server.

You can also assume that no deadline extensions will be given (as per usual really).

4 And To Conclude

Good luck and if you need anything, contact me on joseph dot cordina at um.edu.mt. Even better, make use of the discussion group to discuss ideas and ask questions. You can find a link to the discussion group on my website. Other students might help you out in the solution. Just make sure you do not post any code or given concrete solutions. And do not leave the assignment to the last week since you will definitely not make it in time!! Do it slowly and the ideas will mature.