

ECE 428/CS 425/CSE 424 Distributed Systems  
Fall 2005, University of Illinois at Urbana-Champaign

## Programming Assignment 0 - Tutorial

Due Date - None

### Overview

The machine programming (MP) part of the course this semester involves building a peer to peer application. We will build this application in stages - each stage will be an individual MP. In all, there will be four to five MPs.

The purpose of this MP is to give you basic information about the projects, as well as get you started on learning to write multithreaded networking code.

### Programming Language for MPs

The **implementation language of choice for all MPs will be C (preferred) or C++** (we will not support Java). The choice of the preferred language C is motivated by the fact that the code templates we will give you are written in C, making it easier for you to focus on the essentials of the coding, and for us to automatically test your code.

If you have never programmed in C before, but have programmed in either Java or C++, one or more of the following tutorials should help you close the gap. If you have never programmed in either C, Java, or C++, please meet the instructor.

1. Steve Holmes' guide to C Programming, <http://www.strath.ac.uk/IT/Docs/Ccourse/>
2. A. D. Marshall's guide to C, <http://www.cs.cf.ac.uk/Dave/C/CE.html>
3. Other online guides for C programming, <http://www.google.com/search?q=c+tutorial>
4. "C programming language", B. W. Kernighan, D. M. Ritchie, Prentice Hall, 2ed, 1988, ISBN: 0131103628.

### Learning to Write Multithreaded Networking Code

All the MPs for this course will require you to write code that can send messages on a network, and most of them will also require you to use multiple threads. If you have not had experience with multithreaded programming or network programming, you should spend some time to figure these out now, so that you do not have trouble with future assignments.

The communication interface of choice for this class is the sockets library. You can choose your own library for threads, yet for most MPs, using `fork()` might suffice.

**Sockets Programming:** Online tutorials on Sockets programming and Unix network programming are given below:

1. "Beej's Guide to Network Programming", <http://www.ecst.csuchico.edu/~beej/guide/net>

2. "Unix network programming", W. R. Stevens, (Addison-Wesley, 3ed, 2002, Vols. 1 and 2 – ISBN: 0130810819 and ISBN: 0131411551 OR Prentice Hall, 1ed, 1990, ISBN: 0139498761).

Reference 1 above is a good place to start - the code examples will help you with the Suggested Activities given later in this MP.

**Multithreaded Code:** For spawning new threads, you should look into the `fork()` system call (actually creates a new process), `clone()` system call and the `pthread` library. Use the Unix man command to find more information about these.

## Suggested Activities

Write two programs: a TCP server and a TCP client. The server should passively wait for clients to connect to it, and the client should initiate a connection to the server. Any useful server needs to be able to give the client information, so you should have your client send the server a request, and have the server respond. You also need to make sure the server is able to handle multiple client requests at the same time by handling each request in a separate thread.

We recommend that you start by looking at some of the sample code in the above tutorials on sockets programming (especially Beej's online tutorial).

If you have not programmed in C before, we suggest you look at the chapters on "Programming with pointers" in the above tutorials. If you do not feel comfortable using pointers, your progress on future projects might be slowed down.

Feel free to post questions about any of these topics on the newsgroup or to come by office hours.

## Laboratory

We will be using the Engineering Workstations Lab. You will find more information about access (including remote access) and installed software at <http://www.ews.uiuc.edu/>.

One suggestion is to use the machines `dclsn[1-30].ews.uiuc.edu`. The required libraries and header files vary from one operating system to the next. The `dclsn` machines are currently running Solaris 8 (Sun OS 5.8) which is different from older versions of Sun OS and different from the operating systems on the HP machines. One implication of this is you need to link to different libraries and include different header files for different systems. Another implication for large systems is that there are multiple versions of the man pages available. Running on the `dclsn` machines, if you are not relying on the standard file `/usr/local/ews/system.cshrc` in your `.cshrc` file you are advised to at least have your environment variable `MANPATH` the same as specified in that file.

Please contact the TA for further questions regarding the laboratory.