Network Lab

Computer Systems and Platforms Lab
Department of Computer Science and Engineering
Seoul National University

Lab Preview

In this Lab we will create a Mcdonald's server:

- Server-side:
  - Should be in charge of the kitchen
  - Handle the clients request
  - Prepare the burgers
  - Send the orders to the clients
- Client-side:
  - Send orders to the kitchen
  - Receive orders back from the kitchen

Introduction

This is the last lab of System Programming

In this lab you will learn:

- How to communicate under TCP/IP network environment.
- How to assure atomicity on critical sections between threads.
- How to limit the number of the clients on listening socket.

Requirements

- Knowledge on how to use pthreads
- Knowledge on how to use sockets

Structure

Files:

- net.c/h, buffer.c/h
  - net.c/h provides a simple networking library for the client and server
  - buffer.c/h implements common constants used by the client & server
- server.c
  - Contains the class used for the server-side
  - Receive information from the client and sends back the order
- client.c
  - Has the class used by the client-side
  - Communicates with the server and receive the completed orders

Net.c

Important Functions:

- *getsocklist:*
  - wrapper for the *getaddrinfo*
  - returns the *addrinfo* struct
  - After use *free* the struct
- *get_line:*
  - Gets strings from the socket
- *put_line:*
  - Sends strings through the socket

client.c

Functions to implement:

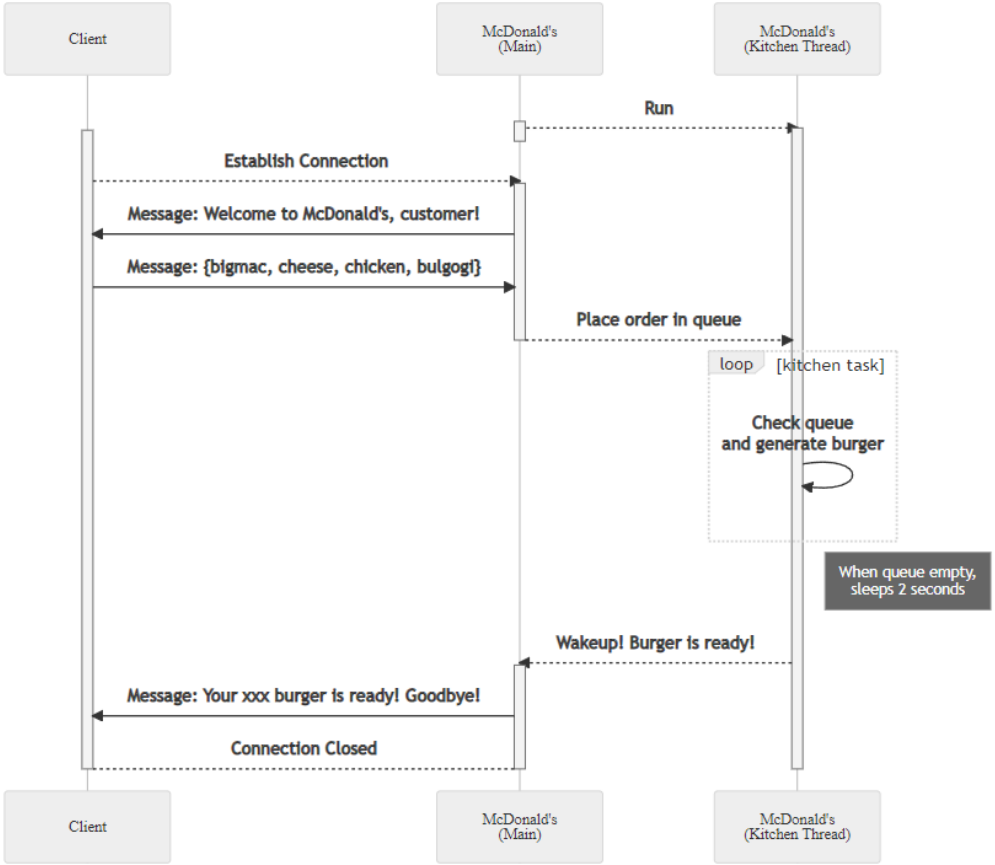
- *main():*
  - create the client thread
  - join the threads
- *thread_task()*
  - connect to the server

server.c

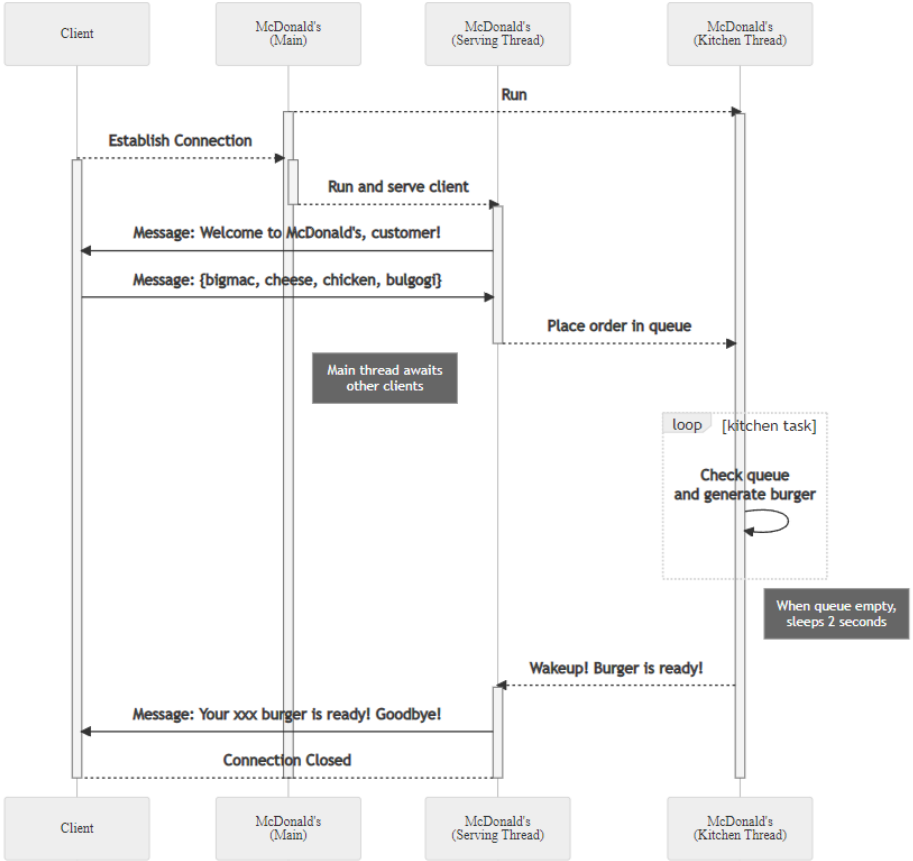
Functions to implement:

- *start_server():*
  - get the socket list and connect it
  - accept clients
- *serve_client()*
  - receive order from the client
  - parse order from the client
  - check if the burguer is available
  - issue the order to the kitchen and wait

Single thread



Multiple threads



Milestones

1. **Send from client and receive in the server**
   - Clients threads well started
   - connection between server and client working
2. **Send from server and receive in client**
   - Connection between server and client working
   - Parsing of the server and send burger working
3. **Make the server multi-threaded**
   - Multiple connections allowed
   - Mutex placed correctly
4. **Limit maximum number of users to 20**
   - Do not allow more than 20 concurrent users (threads)

Hints

Sting operations:

- *asprintf():* print into a string and allocate memory for it at the same time
- *strtok():* tokenize string by delimiter
- *strncmp():* compare two strings by n characters

Hints (2)

Concurrent programming:

- *pthread_create():* create a new thread
- *pthread_detach():* detach a thread
- *pthread_exit():* terminate the calling thread
- *pthread_mutex_init():* create a mutex
- *pthread_mutex_lock():* lock a mutex
- *pthread_mutex_unlock():* unlock a mutex
- *pthread_cond_init():* create a condition variable
- *pthread_cond_wait():* wait on a condition variable
- *pthread_cond_signal():* unblock a thread waiting for a condition variable

Hints(3)

Important structures:

- addrinfo :
  - Contains info related to the socket

Important Dates

Date	Description
Wednesday, November 24	Hand-out
Wednesday, December 1	Lab session 1
Thursday, December 2, 14:00	Milestone 1 submission
Wednesday, December 8	Network Lab session 2
Wednesday, December 8, 23:59	Milestone 2 submission
Friday, December 10, 14:00	Milestone 3 submission
Sunday, December 12, 14:00	Final (milestone 4) submission