# 

Report

*CSS 430 Operating System*

*Project 4*

*By Thong Ton*

# **Implementation Documentation**

### **Overview**

The “shop.h” and “shop.cpp” has been updated for better management. It has more data members because of multiple Barbers, Chairs, Customers. Also, the pthread conditional variables are less far then the original in purpose; only the main shop thread which **pthread\_mutex\_t mutex\_** remains the same in shop.h. The others are moved to Customer and Barber classes.

### **New Classes**

Additional classes are added in the design which are Customer and Barber. This helps the implementation easier because of object oriented programming.

In the Customer and Barber, the implement contains unique ID and conditional pthreads; **cond\_customer\_waiting\_, cond\_customer\_served\_.** In the Barber, the implement. Any thread condition such as calling function **pthread\_cond\_init** will be handled inside the object.

### **New Data structures**

The design implies STL data structures which are a Queues, Maps. Which Queues are used for the available Barbers and waiting Customers. The Maps are used for faster lookup Barbers and Customers by their given unique ID.

#### Use case of Maps

Whenever the Customer comes to the shop, the instance of Customer is created and stored in the Maps which **map<int, Customer\*> customers\_.**

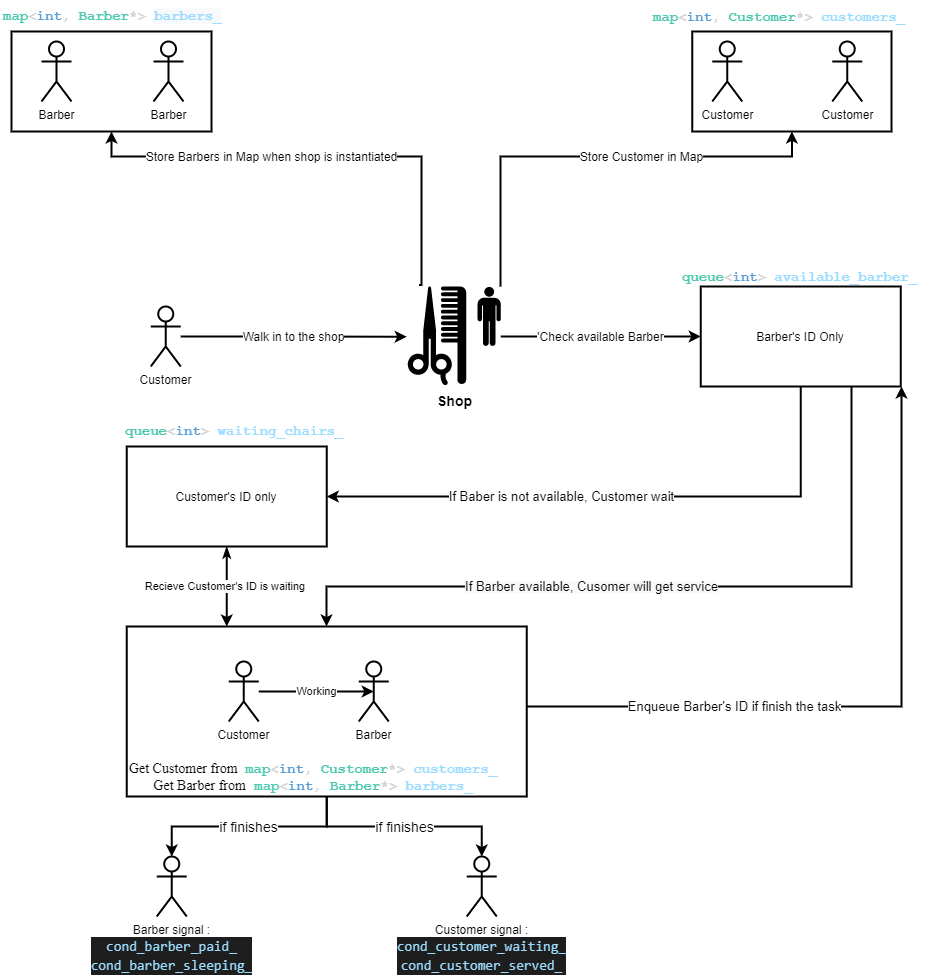
Whenever the Barber is created, it is stored in the Maps.

#### Use case of Queue

Whenever the Customer comes to the shop, the program will look into the **queue<int> available\_barber\_** to require the available Baber.

If the **queue<int> available\_barber\_** is empty, put the Customer into the waiting chair queue which **queue<int> waiting\_chairs\_.**

## **Illustration:** [**Link**](https://drive.google.com/file/d/1aXvsGHZtiEr6gtLWFoyhTzCznXdb_T9z/view?usp=sharing)



# **Discussion**

## **Limitation**

There are some limitations for the design. Because of implementation of pthread and it seems like the numbers of cores and clock skews depend on each machine. The extension and testing could be challenged to make an accurate result.

## **Extension**

The design is possible to make changes and extensions. New classes are added. Also, the Customer and Barber have their own instance. If the project requires more functionality, it is able to make the extent; it is flexible. The design has followed the object oriented programming pattern. This makes the extension more manageable.

## **Discussion step 5**

Approximately how many waiting chairs would be necessary for all 200 customers to be served by 1 barber? Note that depending on the number of cores and clock skews, it may very well require over 60 chairs to accomplish the task.

* For the max chair number # customers who didn't receive a service ~= 30 -> 39”
* So there is not an appropriate number of chairs from 1 -> 60 that can serve all of 200 customers.
* For some reason, the time that I test from csslab5.uwb.edu needs at least 95 chairs to serve all 200 customers.

## **Discussion step 6**

Approximately how many barbers would be necessary for all 200 customers to be served without waiting?

* For one baber the “# customers who didn't receive a service ~= 90 -> 99”
* For two baber the “# customers who didn't receive a service ~= 2 -> 10”
* For three baber the “# customers who didn't receive a service == 0”
* To sum it up, the number needed for serving 200 customers without waiting should be three.

# **Memory Leak**

## Detect Leak:

It is possible to leak memory in the original driver.cpp. The Customer has been assigned to pthread\_joint() to later deallocate the memory. However, the leak possibly comes from the Baber since driver.cpp does not specify any deallocate the point when using pthread.

## Possible Solution:

Because the customer needs to be finished then leaves the shop. Then it is done. However, the barber has finished the task and is waiting for the next task. It is never done literally. Then using pthread\_joint is inappropriate for the Barbers. I use pthread\_detach() for Barbers since I don’t know when the termination point of the Barbers.