Edilberto F. Carrizales Oct 29, 2022

DMV Simulation Design

List of Semaphores Used (including purpose and initial value):

Semaphores	Purpose	Initial value
agentServing	Used by agent to tell the customer when he is being served by them and for customer to wait to be served by an agent	0
customerTicketRequest	Used to determine when a customer wants a ticket and when the information desk has a ticket request to fulfill	0
DMVagent[]	Used to separate the agents 0 and 1 and allow both to take the next available customer and process their exam, photo, and give them a temporary license	{0, 0}
ticketGiven	Used to end the interaction between the customer and information desk by saying that customer got their ticket	0
agentAvailable	Used to tell the customer when an agent is available and to acquire one if it is or wait for wait if it is not	0
announceCustomerTicket	Used to let customer know when a ticket has been announced and to wait if it hasn't	0
customerInWaitingArea	Used so that announcer knows when there is a customer in the waiting area so that it can start announcing	0
customerReadyEyePhoto[]	Used by agent and customer to tell each other when they are ready to begin the eye exam and photo	{0, 0}
customerInAgentLine	Used to tell the announcer when a customer has left	0

	the waiting area and is in the agent line	
customerReadyLicense	Used to determine when a customer is ready for their license, was used to separate eye exam and photo interactions from drivers license	0
customerLeaves	Used by the customer to signal that he has left the dmv and by agent to know when he can accept next customer	0
agentLine	Agent line was used by announcer and customer so that there are always at most 4 people at the agent line, if there aren't, announcer calls another customer to the line, and when a customer leaves he releases the agent line resource to allow another customer	4
infoDeskLine	Creates a line at the info desk for customers coming into the DMV (line is technically infinite but to add realism to the simulation I chose 5)	5
waitingArea	Creates a waiting area for the DMV after they get their ticket from the information desk (the waiting area is supposed to be infinite, so I just chose max num of customers 20)	20
licenseGiven[]	Used to end the interaction between agent and customer by telling a customer that he has received his temporary license	{0, 0}
customerLicenseRequest[]	Used to tie a customer to an agent so that it can keep that agent for the eye, photo and license	{0, 0}

Mutex	will prevent multiple threads from getting mixed up or grabbing same ticket number from information desk	1
eyeAndPhotoTaken[]	Used to tell the customer that the eye exam and the photo have been taken/have finished, and for customer to wait for them to finish	{0, 0}

Other Variables Used:

Other Variables	**purpose**
Int ticket_number	Allow the tracking of the ticket number
Int customerLicenseRequestID[]	Used to give the agent the customer information (customer id) so that they know who they are currently serving

Pseudocode for each function:

(Note: The code was done in java, but I called them wait and signal here)

```
public static class Customer implements Runnable
{
        Int id; //will store the id of the customer

        Public void run()
        {
        //enters the DMV
        Wait(infoDeskLine);
        customerEntersDMV();

        //Get ticket from Information Desk and go to waiting area
        wait(mutex);
        signal(customerTicketRequest);
        wait(ticketGiven);
        customerGetsTicketNumber();
        wait(waitingArea);
        Signal(infoDeskLine);
        signal(mutex);

        //Wait for announcer to call ticket
        signal(customerInWaitingArea);
        Wait(announceCustomerTicket);
```

```
        //Once ticket is called, go to the agent and wait for an agent
        customerMovesToAgentLine();
        signal(waitingArea);
        signal(customerInAgentLine);
        wait(agentAvailable);

        //When an agent is available we want to determine who it is and go to them
        int agent_num; //store the agent

        if (wait(DMVagent[0]) is not available) {
                wait(DMVagent[1]);
                agent_num =1;
        else {
                agent_num = 0;
        }

        //Tell agent who they are, request a license
        customerLicenseRequestID[agent_num] = id;
        signal(customerLicenseRequest[agent_num]);

        //At this point the customer is being served by an agent
        wait (agentServing);
        signal (agentLine);
        customerServedByAgent();

        //Customer will then take eye exam and photo
        signal(customerReadyEyePhoto[agent_num]);
        wait(eyeAndPhotoTaken[agent_num]);
        customerCompletesEyeAndPhoto();

        //Customer is now ready to receive their license and finish the interaction with the patient
        signal(customerReadyLicense);
        wait(licenseGiven[agent_num]);
        customerGetsLicense();

        //customer leaves the DMV
        signal(customerLeaves);

        } //end of void run
} //end of Customer class
```

```
public static class InformationDesk implements Runnable
{
        Public void run()
        {
                informationDeskCreated();
                While (True)
                {
                        //Information Desk waits for a customer to request a ticket number
                        and gives them one

                        Wait(customerTicketRequest);
                        signal(ticketGiven);
                        ticket_number += 1
                }
        } //end of void run
} // end of InformationDesk class


public static class Announcer implements Runnable
{
        Int number = 1; //number to keep track of the announced tickets
        Public void run()
        {
                announcerCreated();
                While (True)
                {

                        //Announcer waits for people to arrive to the waiting and for agent
                        line to have a spot
                        Wait(customerInWaitingArea);
                        Wait(agentLine);

                        //Announcer calls next ticket number
                        announcerCallsNumber();
                         Signal (announceCustomerTicket);

                        //Wait for customer to go in line before next announcement
                        Wait(customerInAgentLine);

                        number +=1;
                }
        } // end of void run
} //end of Announcer class
```

```
public static class Agent implements Runnable
{
        Int id; //number to keep track of the agent id
         Public void run()
        {
                agentCreated();
                While (True)
                {
                        //There are two DMV agents and any of them can grab a customer

                        //Accept the next customer in the agent line
                        Wait(DMVagent[id]);
                        signal(agentAvailable);

                        //Get the customer's driver's license request and their customer id
                        wait(customerLicenseRequest[id]);
                        int customerID = customerLicenseRequestID[id];
                        agentServingCustomer();
                        signal(agentServing);

                        //The agent will ask customer to take their eye exam and photo
                        wait(customerReadyEyePhoto[id]);
                        agentAsksCustomerForEyeAndPhoto();
                        signal(eyeAndPhotoTaken[id]);

                        //Now the driver's license can be given to the customer
                        wait(customerReadyLicense);
                        agentGivesLicenseToCustomer();
                        signal(licenseGiven[id]);

                        //wait for the customer to leave
                        wait(customerLeaves);
                }
        } // end of void run
} //end of Agent class
```