SOFTWARE SPECIFICATION

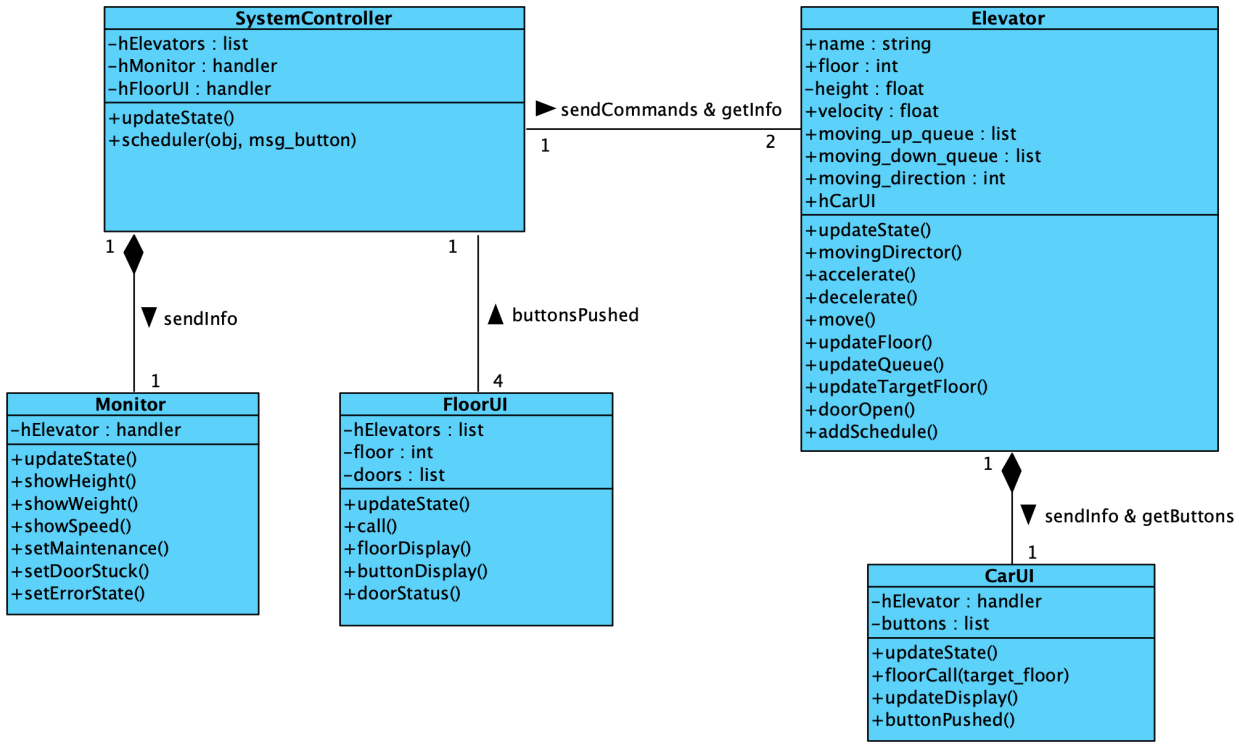
Elevator System

Team 1
Author: Chenyang Zhang

Table of Contents

System Architecture

The elevator system is controlled by a *SystemController*, receiving button information from each *FloorUI* and *Elevator*, and sending information and commands to the *Monitor* and *Elevator*. Each Elevator has a CarUI, with which users can observe the elevator status and select floors to go. The architecture of the elevator system is shown below.



Software Specification

S1: System Controller Implementation

S1.1: Scheduler

S1.1.1: Build hierarchy

1. If the elevator is still and idle, hierarchy = 0
2. If the elevator is moving up or door opened:
    a) If calling floor > elevator's height:
        i. If the calling direction is up, hierarchy=1
        ii. If the calling direction is down, hierarchy=2
    b) If calling floor <= elevator's height:
        i. If the calling direction is up, hierarchy=4
        ii. If the calling direction is down, hierarchy=3
3. If the elevator is moving down or door opened:
    a) If calling floor > elevator's height:
        i. If the calling direction is up, hierarchy=3
        ii. If the calling direction is down, hierarchy=4
    b) If calling floor <= elevator's height:
        i. If the calling direction is up, hierarchy=2
        ii. If the calling direction is down, hierarchy=1
4. If the call is related to BF, only elevator2 should be called:
    a) If F1 is calling down, elevator2's hierarchy=0
    b) If FB is calling up, elevator2's hierarchy=0

S1.1.2: Compare hierarchy

1. If Hie_elevator1 < Hie_elevator2, elevator1 is dispatched
2. If Hie_elevator1 > Hie_elevator2, elevator2 is dispatched
3. If Hie_elevator1 == Hie_elevator2, compare distance to the calling floor:
    a) If Dis_elevator1 <= Dis_elevator2, elevator1 is dispatched
    b) If Dis_elevator1 > Dis_elevator2, elevator2 is dispatched

S2: Elevator Implementation

S2.1: Move

S2.1.1: Check speed

1. If the elevator distance less than the braking distance to the target floor, the elevator should slow down, i.e. speed += 0.1.
2. Elseif the elevator's speed is less than the cruising speed, the elevator should slow down, i.e. speed -= 0.1

S2.1.2: Update height

1. If moving_direction == 1:
    a) If height + speed > height_limit_max, height = height
    b) Else, height += speed
2. If moving_direction == -1:
    a) If height - speed < height_limit_min, height = height

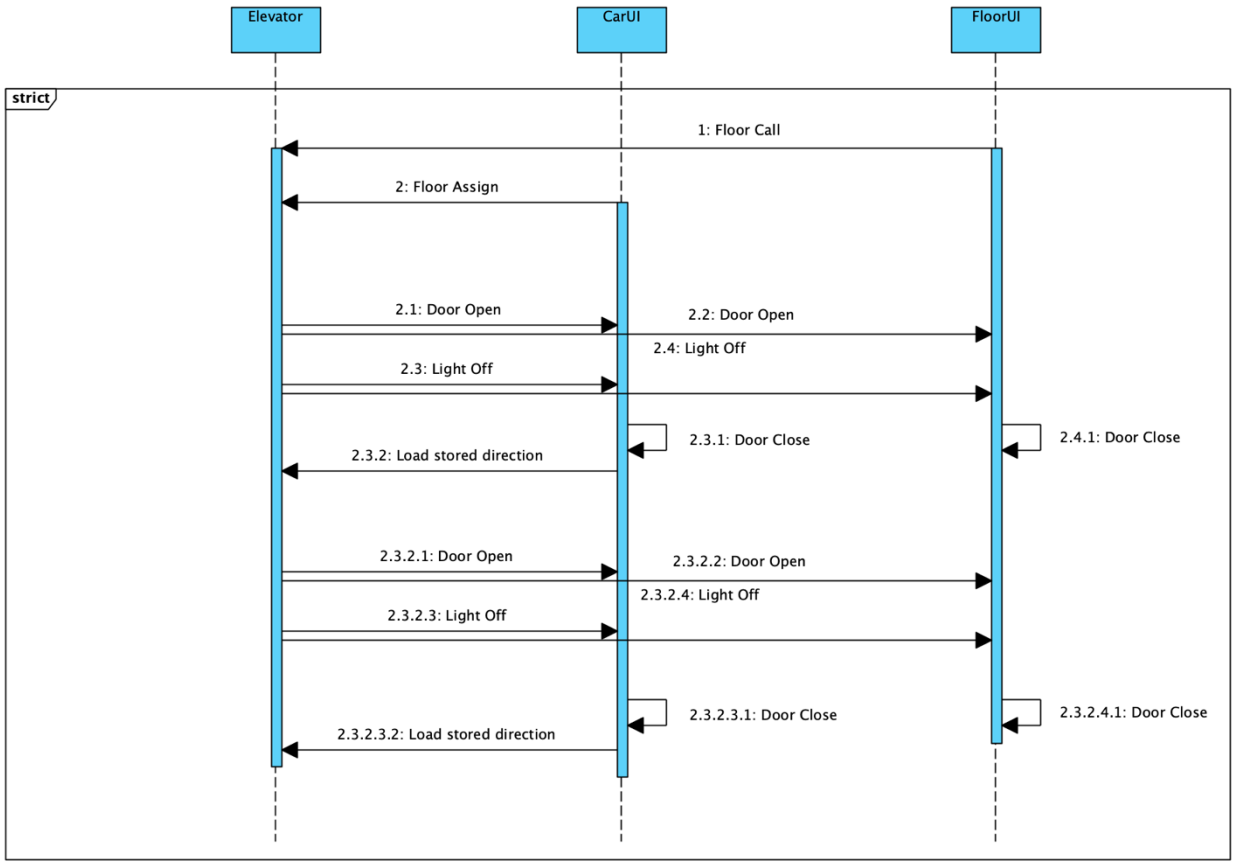
b) Else, height -= speed

3. If moving_direction == 0, height = height

S2.1.3: Check direction

1. If moving_direction == 1, there are 4 cases that the elevator should stop moving, i.e. moving_direction = 0:
   a) Both moving_up_queue and moving_down_queue are empty.
   b) Moving_up_queue is empty, and the height of the elevator > max(moving_down_queue).
   c) Moving_down_queue is empty, and the height of the elevator > max(moving_up_queue).
   d) The height of the elevator > max(moving_down_queue) and > max(moving_up_queue).
2. If moving_direction == -1, there are 4 cases that the elevator should stop moving, i.e. moving_direction = 0:
   a) Both moving_up_queue and moving_down_queue are empty.
   b) Moving_up_queue is empty, and the height of the elevator < min(moving_down_queue).
   c) Moving_down_queue is empty, and the height of the elevator < min(moving_up_queue).
   d) The height of the elevator < min(moving_down_queue) and < min(moving_up_queue).
3. If moving_direction == 0, the elevator should be redirecting
   a) If moving_up_queue is not empty, moving to the first floor in the moving_up_queue.
   b) Elseif moving_down_queue is not empty, moving to the first floor in the moving_down_queue.
   c) Else, the elevator keeps idel.

S2.2: Update Queue



S2.2.1: Stop during moving up

1. If moving_direction = 1 and moving_up_queue is not empty:
    a) If the elevator's height is equal to the first element in the moving_up_queue:
        i. Remove the first element in the moving_up_queue
        ii. Store the moving_direction
        iii. CarUI & FloorUI open the door
        iv. CarUI / FloorUI Light off

S2.2.2: Stop during moving down

1. If moving_direction = -1 and moving_down_queue is not empty:
    a) If the elevator's height is equal to the first element in the moving_down_queue:
        i. Remove the first element in the moving_down_queue
        ii. Store the moving_direction
        iii. CarUI & FloorUI open the door
        iv. CarUI / FloorUI Light off

S2.2.3: Stop just at that floor

1. If moving_direction = 0 and moving_up_queue is not empty:
    a) If the elevator's height is equal to the first element in the moving_up_queue:
        i. Remove the first element in the moving_up_queue
        ii. CarUI & FloorUI open the door
        iii. CarUI / FloorUI Light off
2. Elseif moving_direction = 0 and moving_down_queue is not empty:
    a) If the elevator's height is equal to the first element in the moving_down_queue:
        i. Remove the first element in the moving_down_queue
        ii. CarUI & FloorUI open the door
        iii. CarUI / FloorUI Light off

S2.2.4: Load stored queue

1. If moving_direction = 0 and moving_up_queue is empty:
    a) moving_up_queue = moving_up_queue_store.
    b) Clear moving_up_queue_store.
2. If moving_direction = 0 and moving_down_queue is empty:
    a) moving_down_queue = moving_down_queue_store.
    b) Clear moving_down_queue_store.

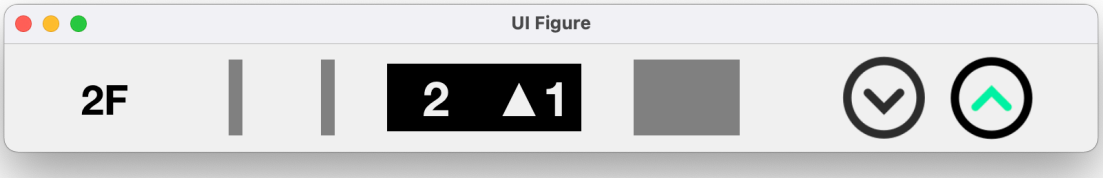
S2.3: Add Schedule

1. If the calling direction is 'up':
    a) If the elevator's height > the height of the calling floor:
        i. Add the calling floor to the moving_up_queue_store
        ii. Sort the moving_up_queue_store in ascend order
    b) Else, if the elevator's height <= the height of the calling floor:
        i. Add the calling floor to the moving_up_queue
        ii. Sort the moving_up_queue in ascend order
2. If the calling direction is 'down':
    a) If the elevator's height < the height of the calling floor:

i. Add the calling floor to the moving_down_queue_store
ii. Sort the moving_down_queue_store in descend order

b) Else, if the elevator's height >= the height of the calling floor:
i. Add the calling floor to the moving_down_queue in descend order
ii. Sort the moving_down_queue

S3: FloorUI Implementation



S3.1: Update Display

S3.1.1: Floor display

1. If the elevator is at FB, the display should be 'B';
2. If the elevator is at F1, the display should be '1';
3. If the elevator is at F2, the display should be '2';
4. If the elevator is at F3, the display should be '3'.

S3.1.2: Direction display

1. If the elevator is moving up, the display should be '▲' + *floor display;*
2. If the elevator is moving down, the display should be '▼' + *floor display;*
3. If the elevator is still, the display should be *floor display.*

S3.1.3: Maintenance

1. If the Maintenance mode is ON, the display should be 'M'.

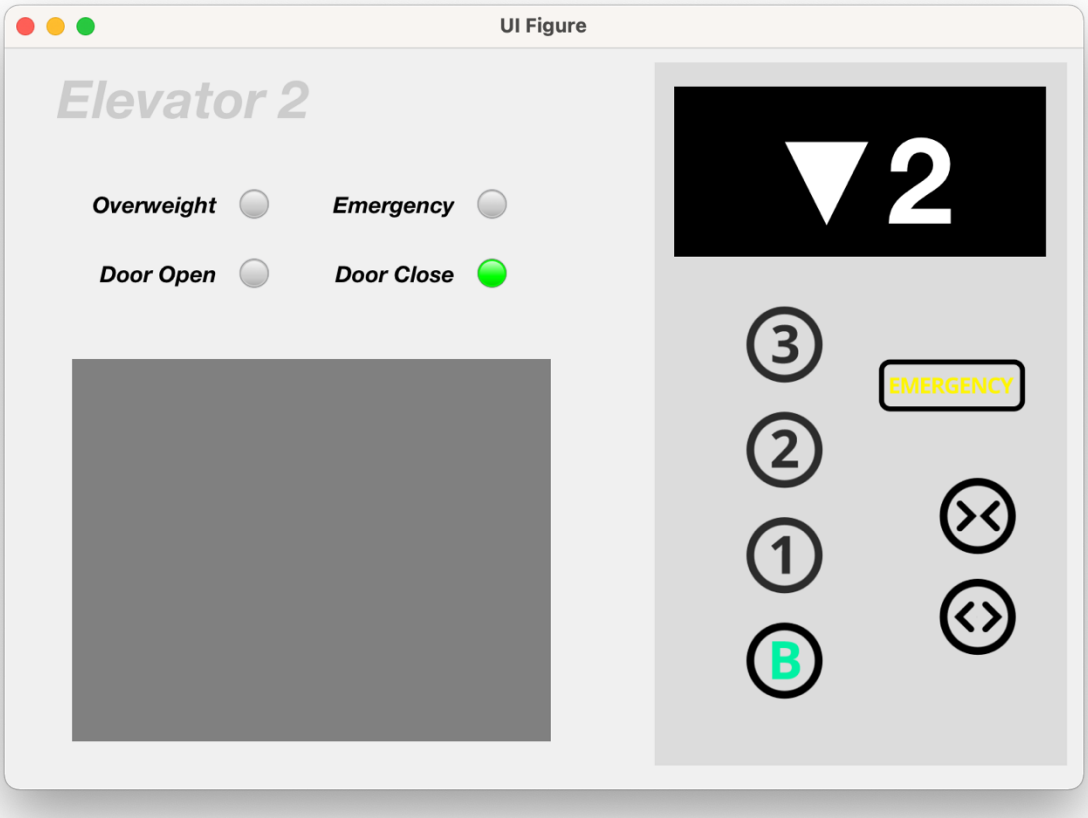
S3.2: Door Status Show

The door has 4 states: close, opening, open, and closing. In FloorUI, the door status cannot be controller independently.

S3.3: Floor Call

1. If the 'Up' button is pressed:
   a) The color should turn into green
   b) An (_F, up) call should been made
2. If the 'Down' button is pressed:
   a) The color should turn into green
   b) An (_F, down) call should been made

S4: CarUI Implementation



S4.1: Update Display

S4.1.1: Floor display

1. If the elevator is at FB, the display should be 'B';
2. If the elevator is at F1, the display should be '1';
3. If the elevator is at F2, the display should be '2';
4. If the elevator is at F3, the display should be '3'.

S4.1.2: Direction display

1. If the elevator is moving up, the display should be '▲' + *floor display;*
2. If the elevator is moving down, the display should be '▼' + *floor display;*
3. If the elevator is still, the display should be *floor display.*

S4.1.3: Button display

1. If a floor button is pressed, it should be lightened until this floor is arrived.
2. If the 'door close' button is pressed:
   a) If door_status == 'open' or door_status == 'opening', the door should be set to closing status;
   b) If door_status == 'close' or door_status == 'closing', the door should keep its status;
   c) If the elevator is moving, this button should be invalid.
3. If the 'door open' button is pressed:
   a) If door_status == 'close' or door_status == 'closing', the door should be set to opening status;

b) If door_status == ‘open’ or door_status == ‘opening’, the door should keep its status;
c) If the elevator is moving, this button should be invalid.

4. If the ‘EMERGENCY’ button is pressed:
   a) the elevator should decelerate and stop immediately;
   b) the color of button should be set to red.

S4.1.4: Signal light display

1. ‘Overweight’ should be lightened when the elevator is overweight;
2. ‘Emergency’ should be lightened when EMERGENCY button is pressed;
3. ‘Door Open’ should be lightened red when the door is opening;
4. ‘Door Open’ should be lightened green when the door is open;
5. ‘Door Close’ should be lightened red when the door is closing;
6. ‘Door Close’ should be lightened green when the door is closed;

S4.1.5: Door display

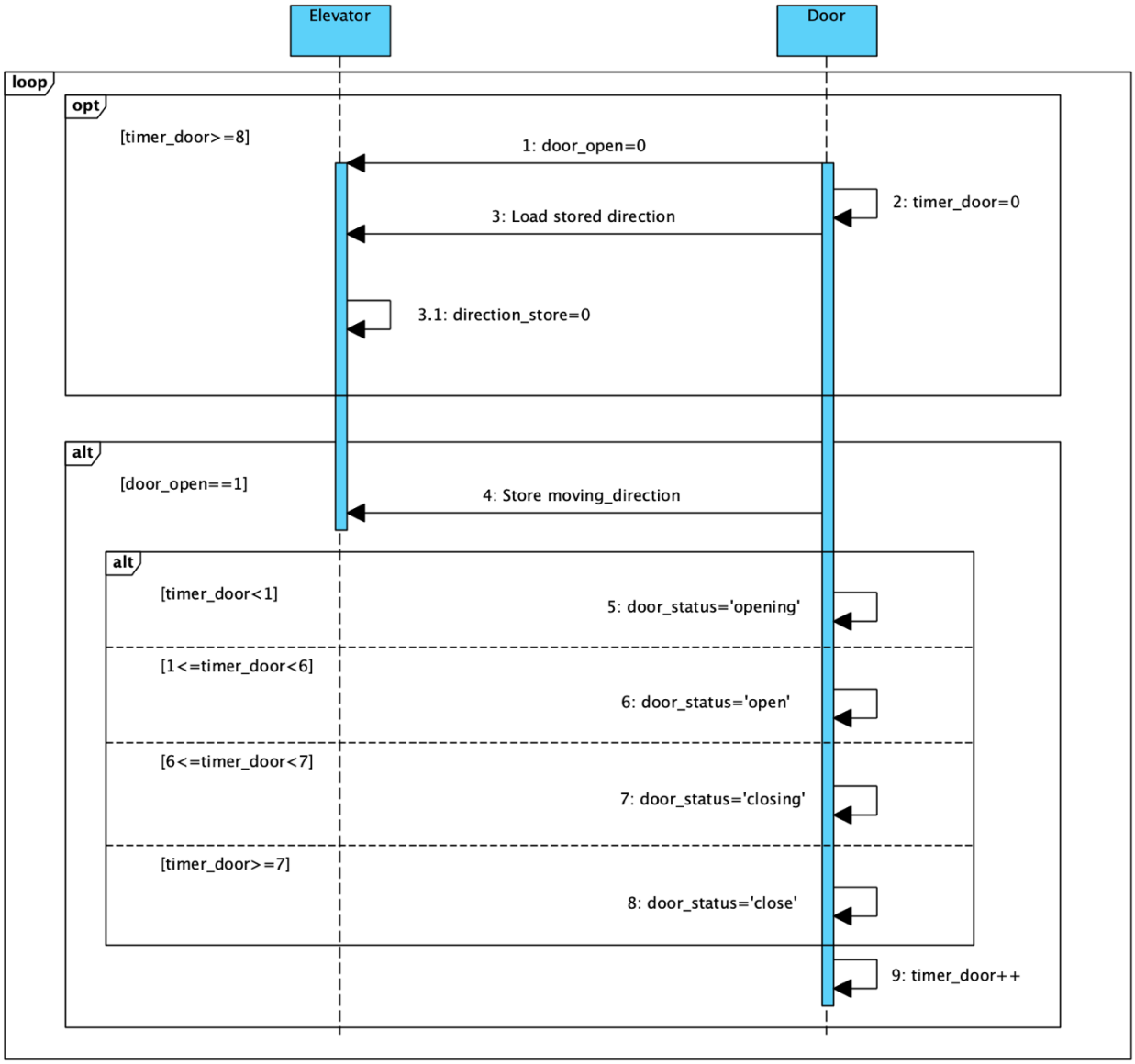
The door has 4 states: close, opening, open, and closing.

S4.2: Door Control

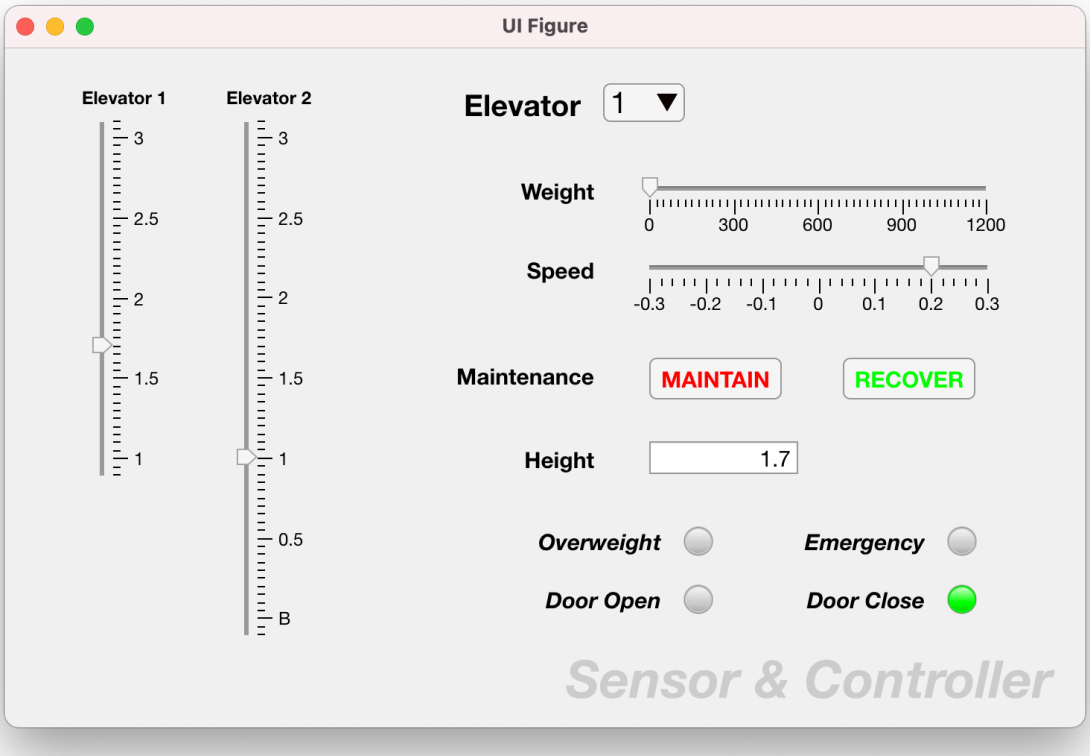
1. If timer_door >= 8:
   a) The door should be closed, i.e. door_open == 0;
   b) Timer_door should be set to 0;
   c) Load stored direction direction_store to the moving_direction;
   d) Set direction_store to 0;
   e) Redirecting
2. If foor_open == 1:
   a) Store moving_direction to direction_store;
   b) If timer_door < 1: set door_status to ‘opening’;
   c) If 1 <= timer_door < 6: set door_status to ‘open’;
   d) If 6 <= timer_door < 7: set door_status to ‘closing’;
   e) If timer_door >= 7: set door_status to ‘close’
   f) Timer_door += 1;

S4.3: Floor Assign

1. If the target_floor user selected <= the height of the elevator:
   a) Add the calling floor to the moving_down_queue
   b) Sort the moving_down_queue in descend order
2. If the target_floor user selected > the height of the elevator:
   a) Add the calling floor to the moving_up_queue
   b) Sort the moving_up_queue in descend order


Elevator
Door
loop
opt
[timer_door>=8]
1: door_open=0
2: timer_door=0
3: Load stored direction
3.1: direction_store=0
alt
[door_open==1]
4: Store moving_direction
alt
[timer_door<1]
5: door_status='opening'
[1<=timer_door<6]
6: door_status='open'
[6<=timer_door<7]
7: door_status='closing'
[timer_door>=7]
8: door_status='close'
9: timer_door++

S5: Monitor



S5.1: Show Speed

1. If moving_direction==1: speed = velocity;
2. If moving_direction==-1: speed = -velocity;
3. If moving_direction==0: speed = 0;

S5.2: Signal light display

1. ‘Overweight’ should be lightened when overweight==1;
2. ‘Emergency’ should be lightened when emergency==1;
3. ‘Door Open’ should be lightened red when the door_status is ‘opening’;
4. ‘Door Open’ should be lightened green when the door_status is ‘open’;
5. ‘Door Close’ should be lightened red when the door_status is ‘closing’;
6. ‘Door Close’ should be lightened green when the door_status is ‘closed’;