SYSC3303
Real Time Concurrent Systems

Elevator Control System
Final Project Report

Group B1-G1

Joshua Robson, 101195802
Liam Kavanagh, 101182915
Matthew Huybregts, 101185221
Sean Pruss, 101189970
Abed Qubbaj, 101205030

TA: Igor Bogdanov

Friday April 5th, 2024

Table of Contents

1 - Breakdown of Team Member Contributions

Iteration 1:

- Scheduler subsystem: Liam Kavanagh
- Floor subsystem: Matthew Huybregts
- Elevator subsystem: Sean Pruss
- Test code: Joshua Robson
- UML diagrams and README: Abed Qubbaj

Iteration 2:

- Liam Kavanagh: Scheduler State Diagram and Scheduler Coding
- Matthew Huybregts: Floor refactor and Scheduler Coding
- Sean Pruss: Elevator State Diagram and Elevator Coding
- Joshua Robson: JUnit tests
- Abed Qubbaj: Class diagram, sequence diagram and README.md

Iteration 3:

- Liam Kavanagh: Scheduler algorithm and coding
- Matthew Huybregts: Coding, debugging, refactoring
- Sean Pruss: Debugging, also helped out with JUnit tests
- Joshua Robson: JUnit tests
- Abed Qubbaj: Updated diagrams and README.md

Iteration 4:

- Liam Kavanagh: Implementing Door Faults and coding
- Matthew Huybregts: Implementing Timer Class
- Sean Pruss: Timing Diagrams
- Joshua Robson: JUnit tests, Debugging, Coding
- Abed Qubbaj: Updated diagrams and README.md

Iteration 5:

- Liam Kavanagh: Implementing GUI
- Matthew Huybregts: Implementing Capacity Limits
- Sean Pruss: Implementing GUI
- Joshua Robson: JUnit tests, Debugging, Updated UML Diagrams
- Abed Qubbaj: Updated diagrams and README.md

2 - Diagrams of the Elevator System

2.1 UML Class Diagrams

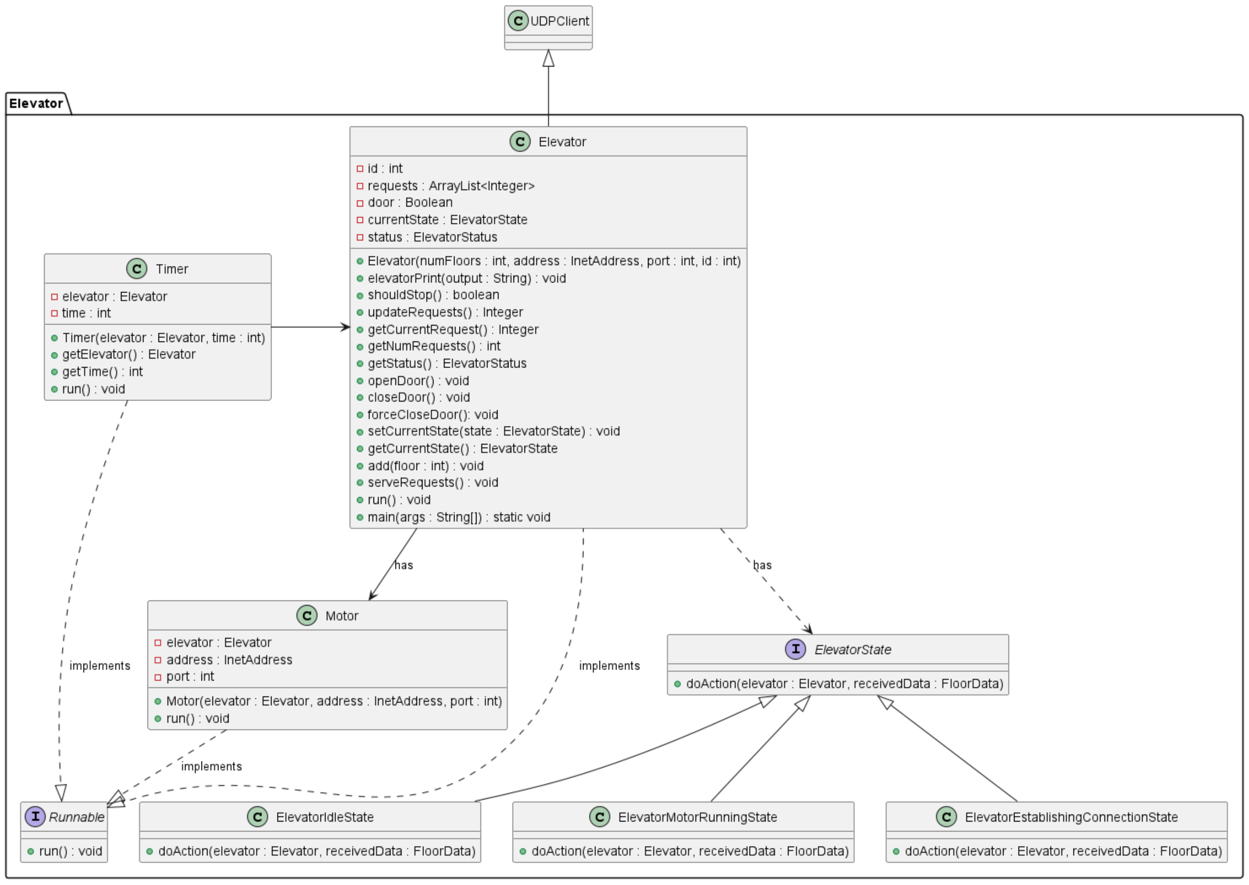

UDPClient
Elevator
Elevator
id : int
requests : ArrayList<Integer>
door : Boolean
currentState : ElevatorState
status : ElevatorStatus
Elevator(numFloors : int, address : InetAddress, port : int, id : int)
elevatorPrint(output : String) : void
shouldStop() : boolean
updateRequests() : Integer
getCurrentRequest() : Integer
getNumRequests() : int
getStatus() : ElevatorStatus
openDoor() : void
closeDoor() : void
forceCloseDoor(): void
setCurrentState(state : ElevatorState) : void
getCurrentState() : ElevatorState
add(floor : int) : void
serveRequests() : void
run() : void
main(args : String[]) : static void
Timer
elevator : Elevator
time : int
Timer(elevator : Elevator, time : int)
getElevator() : Elevator
getTime() : int
run() : void
has
has
Motor
elevator : Elevator
address : InetAddress
port : int
Motor(elevator : Elevator, address : InetAddress, port : int)
run() : void
implements
implements
implements
ElevatorState
doAction(elevator : Elevator, receivedData : FloorData)
Runnable
run() : void
ElevatorIdleState
doAction(elevator : Elevator, receivedData : FloorData)
ElevatorMotorRunningState
doAction(elevator : Elevator, receivedData : FloorData)
ElevatorEstablishingConnectionState
doAction(elevator : Elevator, receivedData : FloorData)

Figure 1: Elevator Class Diagram

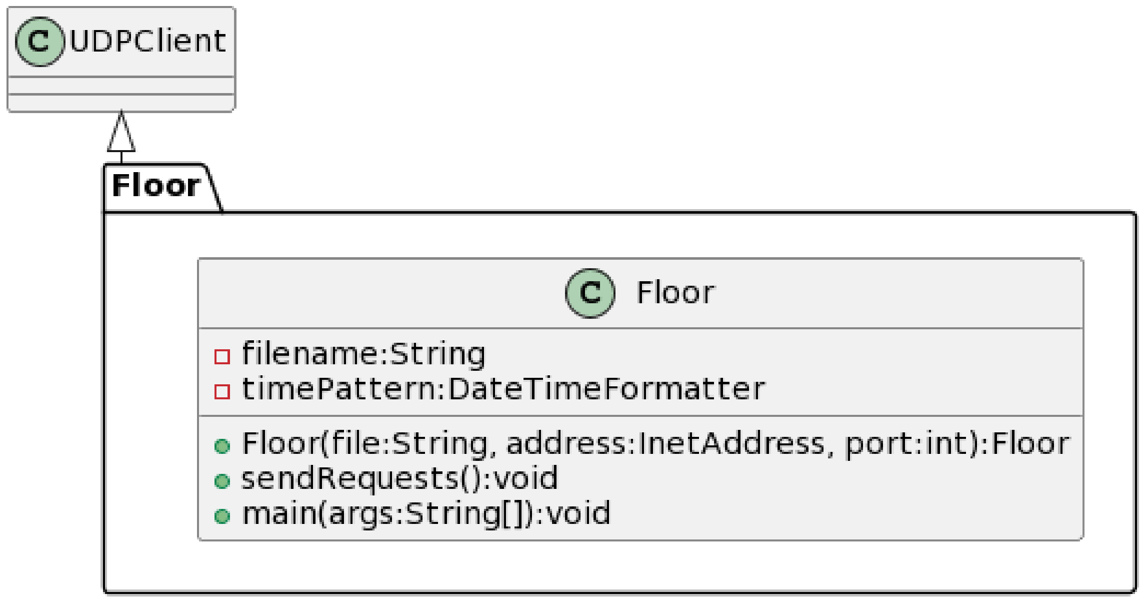

UDPClient
Floor
Floor
filename:String
timePattern:DateTimeFormatter
Floor(file:String, address:InetAddress, port:int):Floor
sendRequests():void
main(args:String[]):void

Figure 2: Floor Subsystem Diagram

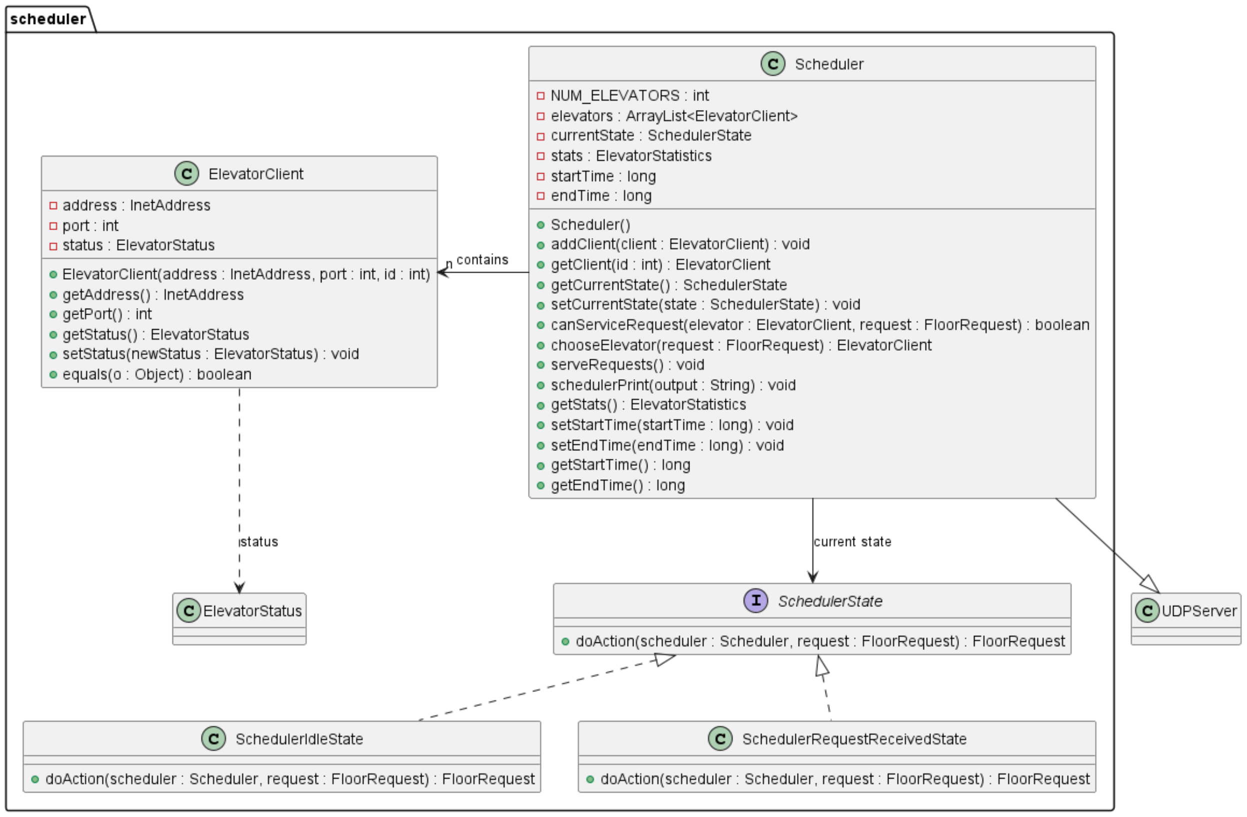

scheduler
Scheduler
NUM_ELEVATORS : int
elevators : ArrayList<ElevatorClient>
currentState : SchedulerState
stats : ElevatorStatistics
startTime : long
endTime : long
Scheduler()
addClient(client : ElevatorClient) : void
getClient(id : int) : ElevatorClient
getCurrentState() : SchedulerState
setCurrentState(state : SchedulerState) : void
canServiceRequest(elevator : ElevatorClient, request : FloorRequest) : boolean
chooseElevator(request : FloorRequest) : ElevatorClient
serveRequests() : void
schedulerPrint(output : String) : void
getStats() : ElevatorStatistics
setStartTime(startTime : long) : void
setEndTime(endTime : long) : void
getStartTime() : long
getEndTime() : long
ElevatorClient
address : InetAddress
port : int
status : ElevatorStatus
ElevatorClient(address : InetAddress, port : int, id : int)
getAddress() : InetAddress
getPort() : int
getStatus() : ElevatorStatus
setStatus(newStatus : ElevatorStatus) : void
equals(o : Object) : boolean
n
contains
status
current state
ElevatorStatus
SchedulerState
doAction(scheduler : Scheduler, request : FloorRequest) : FloorRequest
UDPServer
SchedulerIdleState
doAction(scheduler : Scheduler, request : FloorRequest) : FloorRequest
SchedulerRequestReceivedState
doAction(scheduler : Scheduler, request : FloorRequest) : FloorRequest

Figure 3: Scheduler Subsystem Diagram

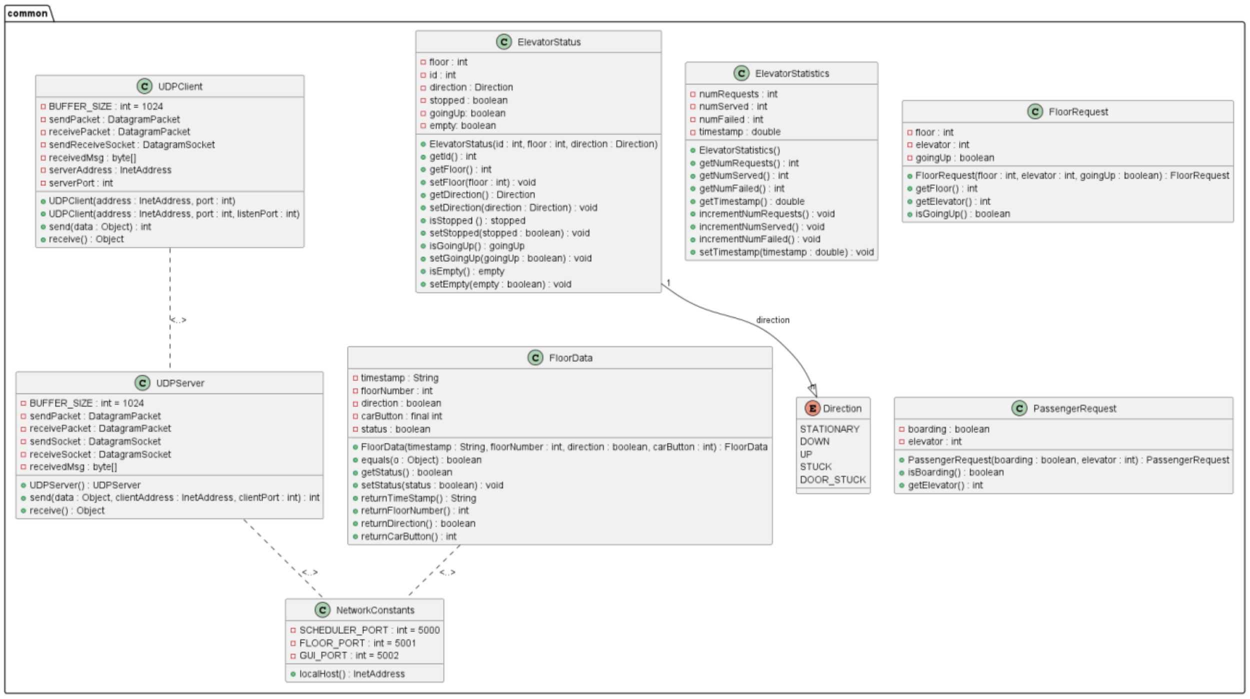

common
UDPClient
BUFFER_SIZE : int = 1024
sendPacket : DatagramPacket
receivePacket : DatagramPacket
sendReceiveSocket : DatagramSocket
receivedMsg : byte[]
serverAddress : InetAddress
serverPort : int
UDPClient(address : InetAddress, port : int)
UDPClient(address : InetAddress, port : int, listenPort : int)
send(data : Object) : int
receive() : Object
ElevatorStatus
floor : int
id : int
direction : Direction
stopped : boolean
goingUp: boolean
empty: boolean
ElevatorStatus(id : int, floor : int, direction : Direction)
getId() : int
getFloor() : int
setFloor(floor : int) : void
getDirection() : Direction
setDirection(direction : Direction) : void
isStopped () : stopped
setStopped(stopped : boolean) : void
isGoingUp() : goingUp
setGoingUp(goingUp : boolean) : void
isEmpty() : empty
setEmpty(empty : boolean) : void
ElevatorStatistics
numRequests : int
numServed : int
numFailed : int
timestamp : double
ElevatorStatistics()
getNumRequests() : int
getNumServed() : int
getNumFailed() : int
getTimestamp() : double
incrementNumRequests() : void
incrementNumServed() : void
incrementNumFailed() : void
setTimestamp(timestamp : double) : void
FloorRequest
floor : int
elevator : int
goingUp : boolean
FloorRequest(floor : int, elevator : int, goingUp : boolean) : FloorRequest
getFloor() : int
getElevator() : int
isGoingUp() : boolean
<..>
1
direction
UDPServer
BUFFER_SIZE : int = 1024
sendPacket : DatagramPacket
receivePacket : DatagramPacket
sendSocket : DatagramSocket
receiveSocket : DatagramSocket
receivedMsg : byte[]
UDPServer() : UDPServer
send(data : Object, clientAddress : InetAddress, clientPort : int) : int
receive() : Object
FloorData
timestamp : String
floorNumber : int
direction : boolean
carButton : final int
status : boolean
FloorData(timestamp : String, floorNumber : int, direction : boolean, carButton : int) : FloorData
equals(o : Object) : boolean
getStatus() : boolean
setStatus(status : boolean) : void
returnTimeStamp() : String
returnFloorNumber() : int
returnDirection() : boolean
returnCarButton() : int
Direction
STATIONARY
DOWN
UP
STUCK
DOOR_STUCK
PassengerRequest
boarding : boolean
elevator : int
PassengerRequest(boarding : boolean, elevator : int) : PassengerRequest
isBoarding() : boolean
getElevator() : int
<..>
<..>
NetworkConstants
SCHEDULER_PORT : int = 5000
FLOOR_PORT : int = 5001
GUI_PORT : int = 5002
localHost() : InetAddress

Figure 4: Common Subsystem Diagram

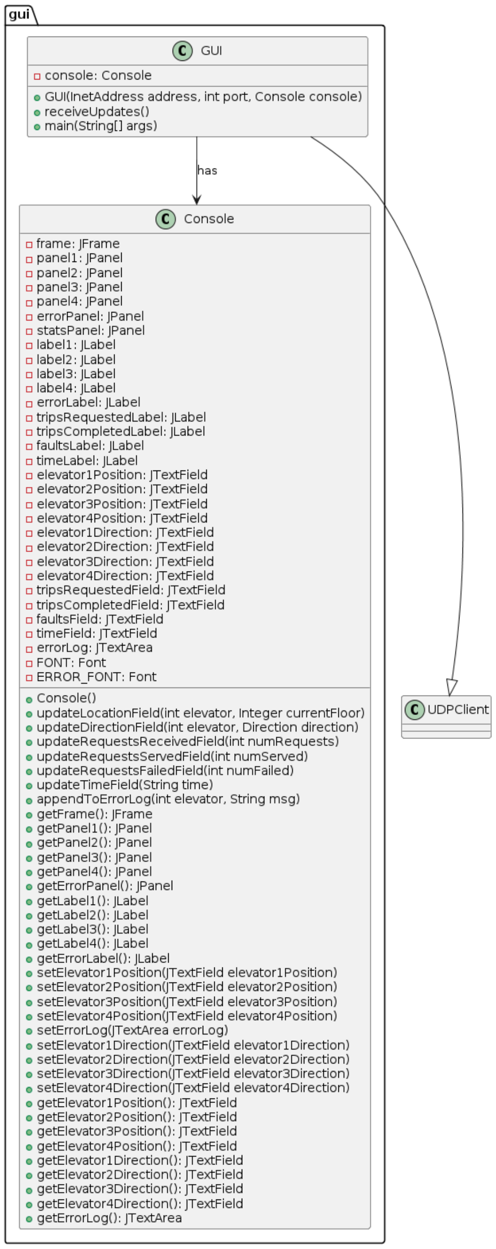
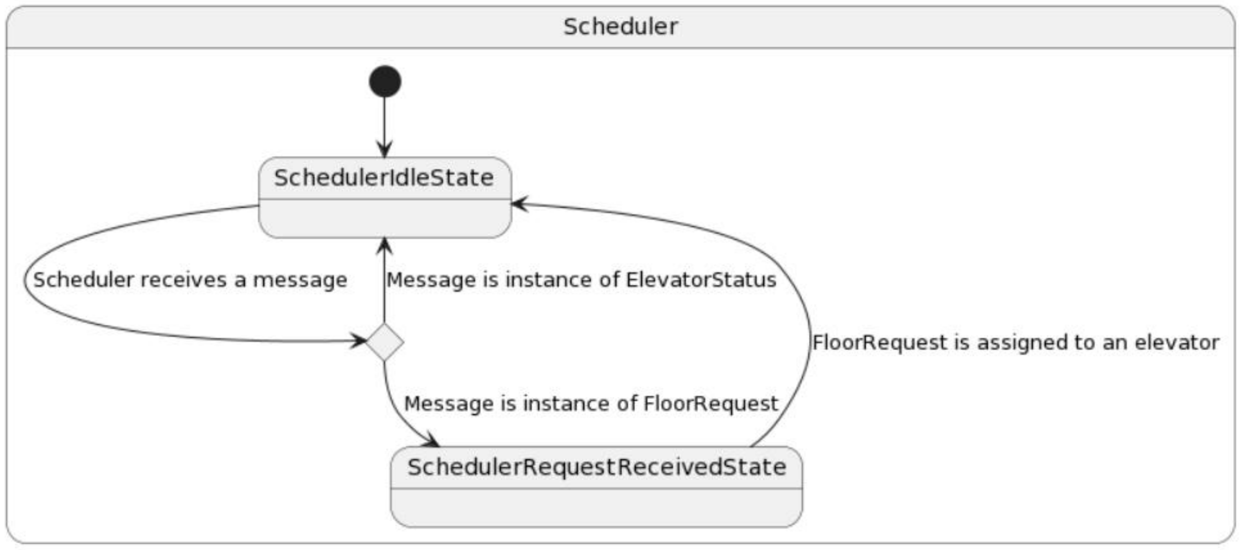

gui
GUI
- console: Console
+ GUI(InetAddress address, int port, Console console)
+ receiveUpdates()
+ main(String[] args)
has
Console
- frame: JFrame
- panel1: JPanel
- panel2: JPanel
- panel3: JPanel
- panel4: JPanel
- errorPanel: JPanel
- statsPanel: JPanel
- label1: JLabel
- label2: JLabel
- label3: JLabel
- label4: JLabel
- errorLabel: JLabel
- tripsRequestedLabel: JLabel
- tripsCompletedLabel: JLabel
- faultsLabel: JLabel
- timeLabel: JLabel
- elevator1Position: JTextField
- elevator2Position: JTextField
- elevator3Position: JTextField
- elevator4Position: JTextField
- elevator1Direction: JTextField
- elevator2Direction: JTextField
- elevator3Direction: JTextField
- elevator4Direction: JTextField
- tripsRequestedField: JTextField
- tripsCompletedField: JTextField
- faultsField: JTextField
- timeField: JTextField
- errorLog: JTextArea
- FONT: Font
- ERROR_FONT: Font
+ Console()
+ updateLocationField(int elevator, Integer currentFloor)
+ updateDirectionField(int elevator, Direction direction)
+ updateRequestsReceivedField(int numRequests)
+ updateRequestsServedField(int numServed)
+ updateRequestsFailedField(int numFailed)
+ updateTimeField(String time)
+ appendToErrorLog(int elevator, String msg)
+ getFrame(): JFrame
+ getPanel1(): JPanel
+ getPanel2(): JPanel
+ getPanel3(): JPanel
+ getPanel4(): JPanel
+ getErrorPanel(): JPanel
+ getLabel1(): JLabel
+ getLabel2(): JLabel
+ getLabel3(): JLabel
+ getLabel4(): JLabel
+ getErrorLabel(): JLabel
+ setElevator1Position(JTextField elevator1Position)
+ setElevator2Position(JTextField elevator2Position)
+ setElevator3Position(JTextField elevator3Position)
+ setElevator4Position(JTextField elevator4Position)
+ setErrorLog(JTextArea errorLog)
+ setElevator1Direction(JTextField elevator1Direction)
+ setElevator2Direction(JTextField elevator2Direction)
+ setElevator3Direction(JTextField elevator3Direction)
+ setElevator4Direction(JTextField elevator4Direction)
+ getElevator1Position(): JTextField
+ getElevator2Position(): JTextField
+ getElevator3Position(): JTextField
+ getElevator4Position(): JTextField
+ getElevator1Direction(): JTextField
+ getElevator2Direction(): JTextField
+ getElevator3Direction(): JTextField
+ getElevator4Direction(): JTextField
+ getErrorLog(): JTextArea
UDPClient

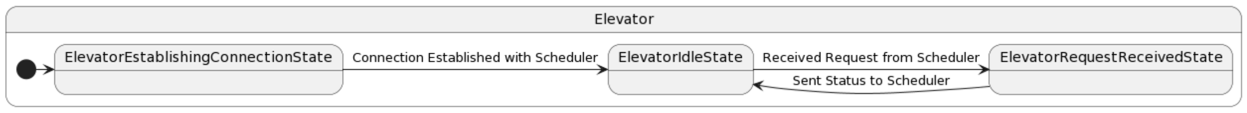
Figure 5: GUI subsystem diagram

2.2 - State Machines



Scheduler
SchedulerIdleState
Scheduler receives a message
Message is instance of ElevatorStatus
FloorRequest is assigned to an elevator
Message is instance of FloorRequest
SchedulerRequestReceivedState

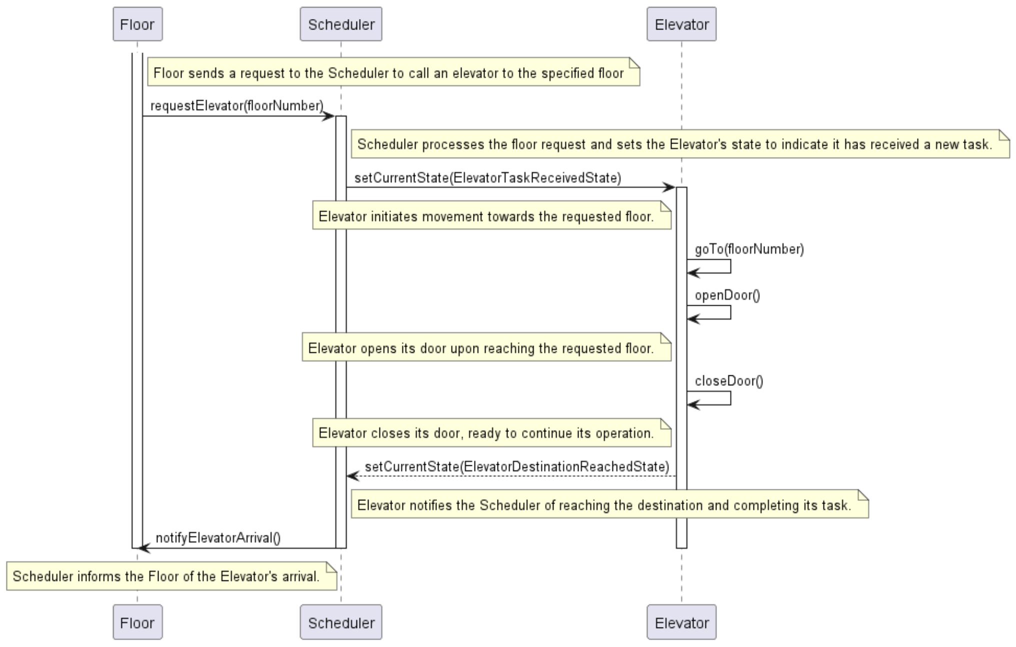
Figure 6: Scheduler State Diagram



Elevator
ElevatorEstablishingConnectionState
Connection Established with Scheduler
ElevatorIdleState
Received Request from Scheduler
ElevatorRequestReceivedState
Sent Status to Scheduler

Figure 7: Elevator State Diagram

2.3 - Sequence Diagrams



Floor
Scheduler
Elevator
Floor sends a request to the Scheduler to call an elevator to the specified floor
requestElevator(floorNumber)
Scheduler processes the floor request and sets the Elevator's state to indicate it has received a new task.
setCurrentState(ElevatorTaskReceivedState)
Elevator initiates movement towards the requested floor.
goTo(floorNumber)
openDoor()
Elevator opens its door upon reaching the requested floor.
closeDoor()
Elevator closes its door, ready to continue its operation.
setCurrentState(ElevatorDestinationReachedState)
Elevator notifies the Scheduler of reaching the destination and completing its task.
notifyElevatorArrival()
Scheduler informs the Floor of the Elevator's arrival.
Floor
Scheduler
Elevator

Figure 8: Elevator System Sequence Diagram

2.4 - Timing Diagrams

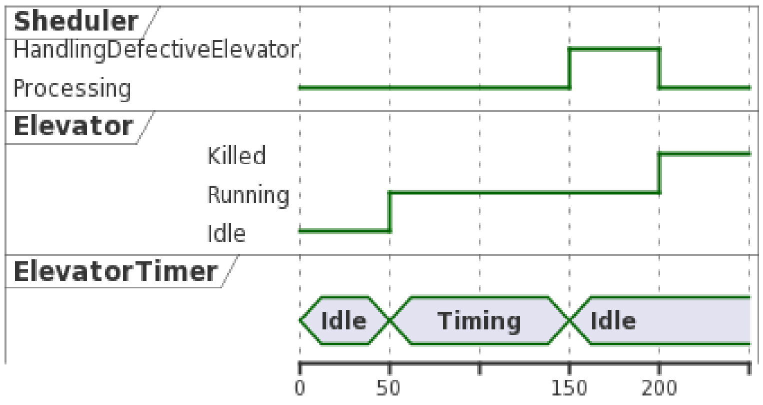


Figure 9: Elevator too slow to reach destination diagram

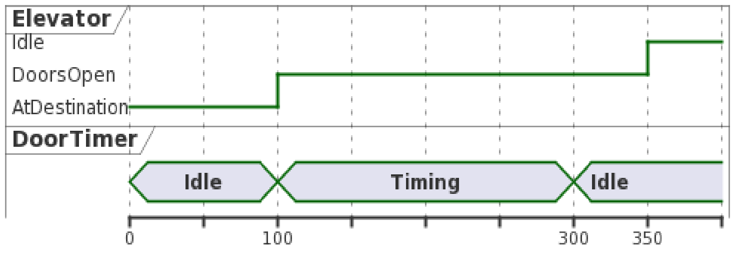


Figure 10: Elevator doors stuck open timing diagram

3 - Detailed Set-Up and Instructions

Requirements

Java SDK `21`, and Language Level set to `Default` SDK in Intellij.

Usage

[!IMPORTANT] Run the files in this order for the system to initialize properly

1. Run `scheduler.Scheduler.java`
2. Run `gui.GUI.java`
3. Run `elevator.Elevator.java`
4. Run `floor.Floor.java`

Input File Format

The floor.Floor subsystem reads input from a file, and each line in the file represents an event. The format is as follows:

[timestamp] [floor_number] [direction (up/down)] [requested_floor]

4 - Result from the measurements

Assumptions:

- There is a distance of 4 meters between floors
- The rate of acceleration is the same as that of the deceleration
- Assume the rate of acceleration is the same no matter the number of floors
- For an adjacent floor, the elevator never maintains a constant speed. It speeds up and slows again

1. **Average Speed (Adjacent Floors)**

   $v = \Delta d/\Delta t$
   $v = 4m/5.225s$
   $v = 0.766m/s$

2. **Average Speed (Multiple Floors)**

   $v = \Delta d/\Delta t$
   $v = 84m/34.37s$
   $v = 2.44m/s$

3. **Acceleration**

   Since we're assuming that the elevator speeds up and then immediately slows down for a single floor, we will assume the maximum velocity is 1.532 m/s (2x the average).

   $a = v/(t/2)$
   $a = 1.532m/s/(5.225/2)s$
   $a = 0.586m/s^2$

4. **Max Speed (Multiple Floors)**

We estimated it takes around 4.73 seconds for the elevator to reach its top speed based on how it felt (may or may not be accurate)

$v = at$

$v = (0.586 m/s^2)(4.73 s)$

$v = 2.77 m/s$

5 - Reflection of our design

5.1 Aspects of the Design We Liked

Our design allowed us to easily implement additional features incrementally. When we approached our design, we wanted to reduce the amount of possible refactoring as much as possible so we could focus on new features. Our project is also well structured, with directories for documentation, source code, and library files which are split into subdirectories such as common, elevator, scheduler, floor, and GUI for the src directory. Our GUI was a separate subsystem of the project, which made it a lot easier to implement the scheduler sending updates to it.

5.2 What Can Be Improved Upon

One improvement we can add is to make the Scheduler and GUI multithreaded. With a multithreaded Scheduler and GUI, we can process requests from multiple elevators and display them at once. This would increase the throughput through increased parallelism. We should also implement a way to deal with the case in which there is no elevator currently available for a passenger. This could be done by notifying the floor when there are no available elevators and having the floor resend the request after a set time and a maximum number of retries for these requests.