SYSC 3303 FINAL PROJECT REPORT

Team 6

Alexander Hum 101180821

Emily Tang 101192064

Nicole Lim 101191181

Nivetha Sivasaravanan 101182962

Rimsha Atif 101187263

TABLE OF CONTENTS

1.0 BREAKDOWN OF RESPONSIBILITIES

The purpose of this section is to describe the breakdown of responsibilities of each team member for each iteration.

1.1 Iteration 1

Table 1 describes the responsibilities of each team member for Iteration 1.

Table 1: Breakdown of Responsibilities of Each Team Member in Iteration 1

Name	Responsibilities
Alexander Hum	Elevator, ElevatorTest, UML Class Diagram, README
Emily Tang	HardwareDevice, HardwareDeviceTest, FloorButton, ElevatorSystem, README
Nicole Lim	Scheduler, SchedulerTest, UML Sequence Diagram, README
Nivetha Sivasaravanan	Floor, FloorTest, README
Rimsha Atif	Scheduler, SchedulerTest, UML Sequence Diagram, README

1.2 Iteration 2

Table 2 describes the responsibilities of each team member for Iteration 2.

Table 2: Breakdown of Responsibilities of Each Team Member in Iteration 2

Name	Responsibilities
Alexander Hum	Scheduler, SchedulerTest, Scheduler State Diagram, README
Emily Tang	ElevatorTest, SchedulerTest, UML Class & Sequence Diagram, README
Nicole Lim	Elevator, ElevatorTest, Elevator State Diagram, README
Nivetha Sivasaravanan	Scheduler, SchedulerTest, Scheduler State Diagram, UML Sequence Diagram, README
Rimsha Atif	Elevator, ElevatorTest, Elevator State Diagram, README

1.3 Iteration 3

Table 3 describes the responsibilities of each team member for Iteration 3.

Table 3: Breakdown of Responsibilities of Each Team Member in Iteration 3

Name	Responsibilities
Alexander Hum	Floor, UML Class Diagram, README
Emily Tang	Scheduler, SchedulerTest, README
Nicole Lim	Scheduler, SchedulerTest, README
Nivetha Sivasaravanan	Elevator, ElevatorTest, README
Rimsha Atif	Floor, ElevatorTest, UML Sequence Diagram, README

1.4 Iteration 4

Table 4 describes the responsibilities of each team member for Iteration 4.

Table 4: Breakdown of Responsibilities of Each Team Member in Iteration 4

Name	Responsibilities
Alexander Hum	Scheduler, SchedulerTest, ElevatorTest, UML Class Diagram, Elevator Timing Diagrams, README
Emily Tang	Elevator, ElevatorStates, ElevatorTest, Elevator State Diagram, UML Class Diagram, README
Nicole Lim	Scheduler, SchedulerTest, UML Class Diagram, Elevator Timing Diagrams, README
Nivetha Sivasaravanan	Floor, HardwareDevice, ElevatorTest, UML Class Diagram, README
Rimsha Atif	Elevator, ElevatorTest, UML Class Diagram, Elevator Timing Diagrams, README

1.5 Iteration 5

Table 5 describes the responsibilities of each team member for Iteration 5.

Table 5: Breakdown of Responsibilities of Each Team Member in Iteration 5

Name	Responsibilities
Alexander Hum	Elevator, ElevatorSystemUI, ElevatorTest, ElevatorSystemView, UML Class Diagram, README
Emily Tang	Elevator, Scheduler, ElevatorTest, ElevatorSystemLogger, HardwareDevice, UML Class Diagram, README
Nicole Lim	Elevator, ElevatorSystemUI, SchedulerStates, ElevatorSystemView, UML Class Diagram, README
Nivetha Sivasaravanan	Elevator, HardwareDevice, ElevatorTest, UML Class Diagram, README
Rimsha Atif	Elevator, HardwareDevice, UML Class Diagram, README

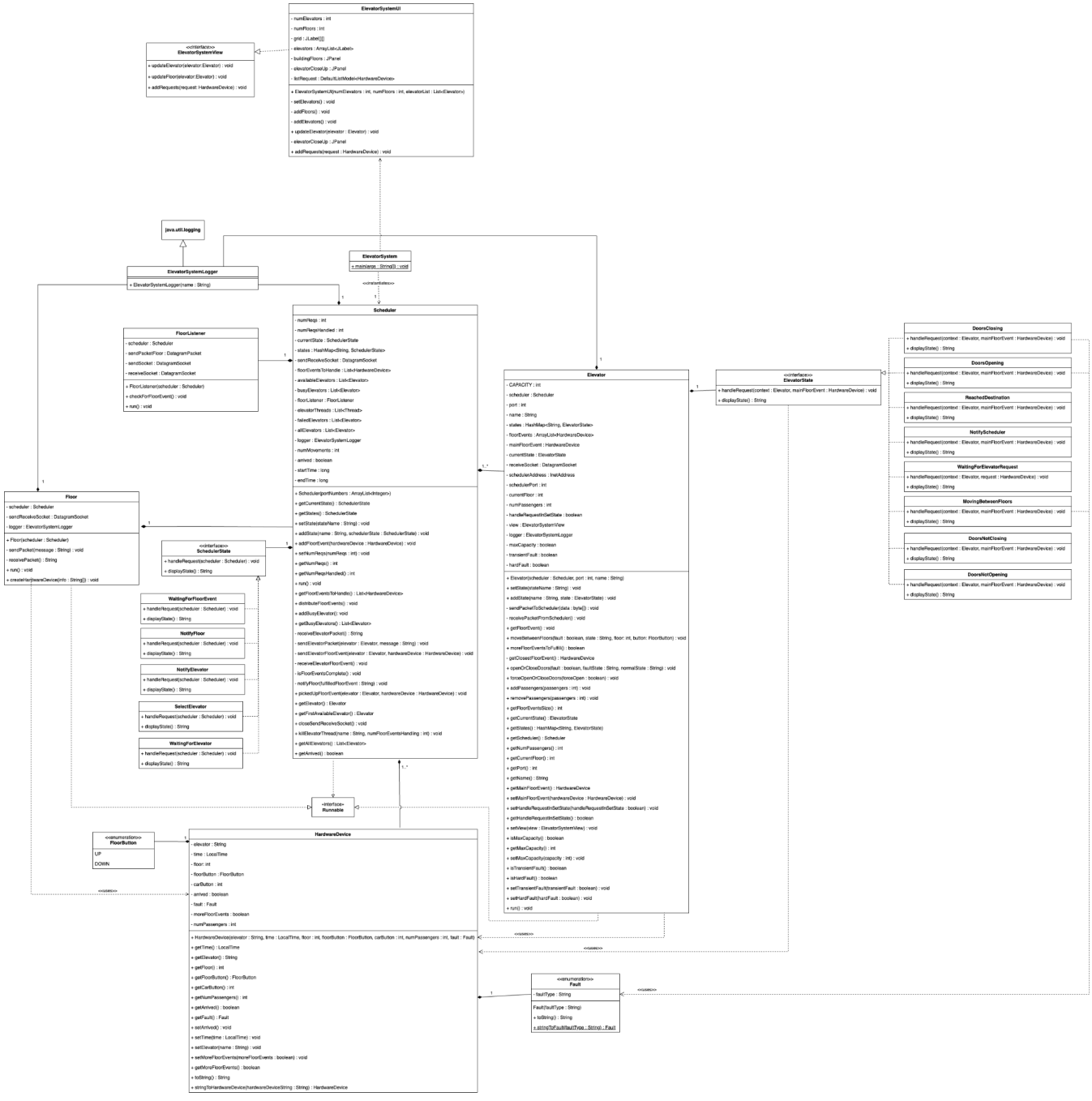
2.0 DIAGRAMS

This section contains the UML Class, Scheduler State Machine, UML Sequence, and Scheduler Timing diagrams for the ElevatorSystem.

2.1 UML Class Diagram

Figure 1 illustrates the UML Class Diagram of the ElevatorSystem.

Figure 1: UML Class Diagram of the ElevatorSystem



2.2 Scheduler State Machine Diagram

Figure 2 illustrates the state machine diagram of the Scheduler in the ElevatorSystem.

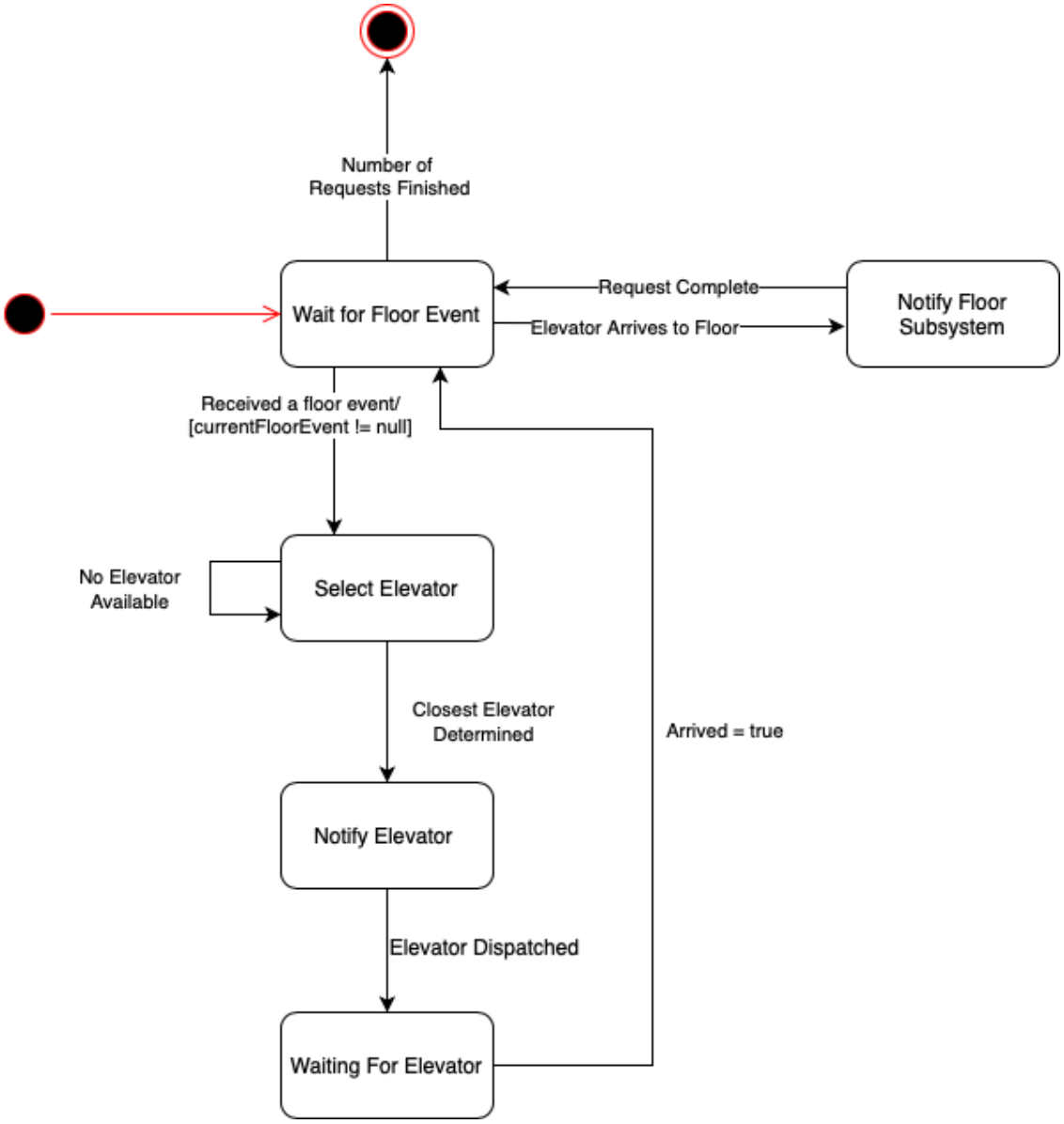
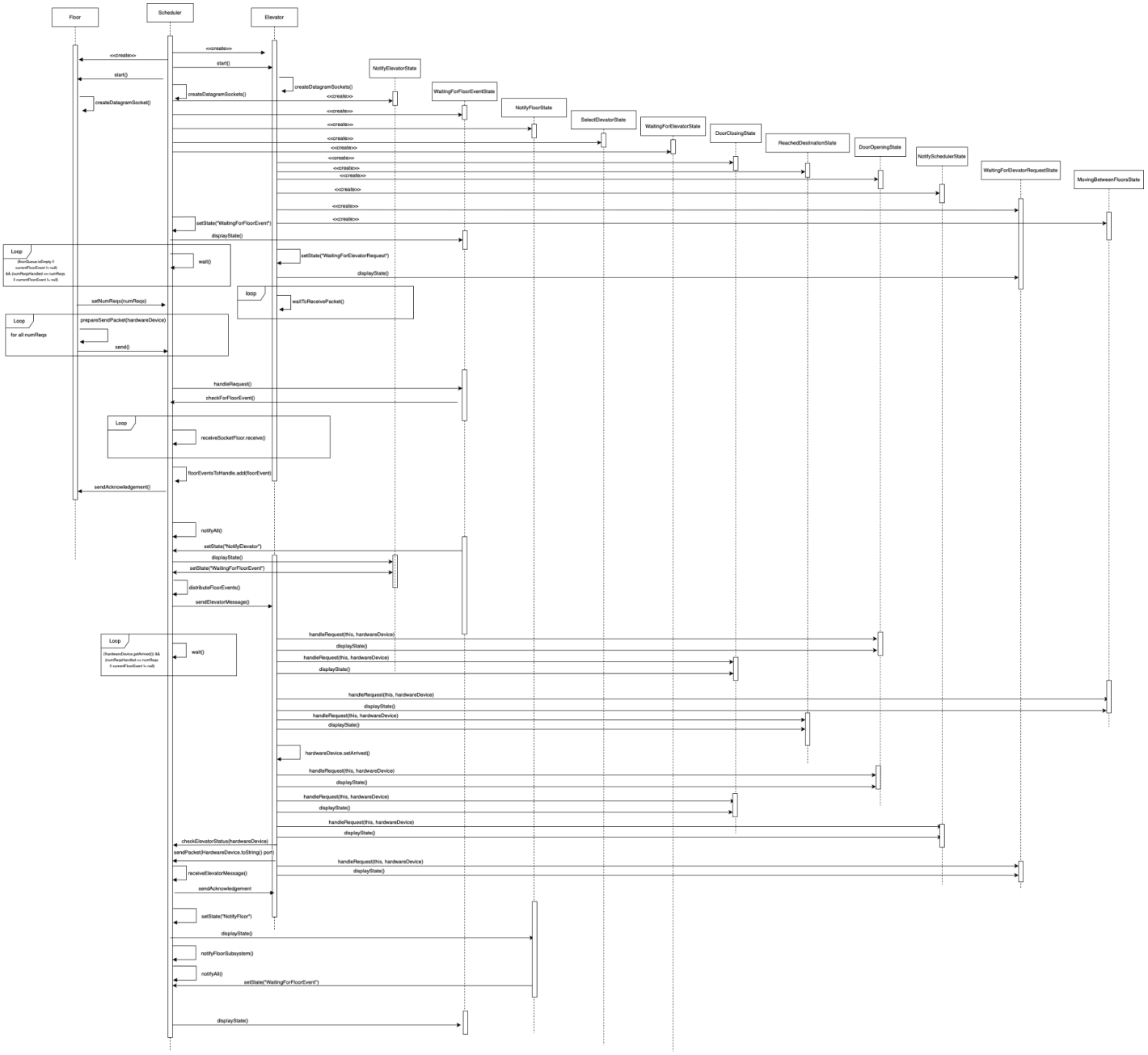


Figure 2: State Machine Diagram of the Scheduler in the ElevatorSystem

2.3 UML Sequence Diagram

Figure 3 illustrates the UML Sequence Diagram of the ElevatorSystem.

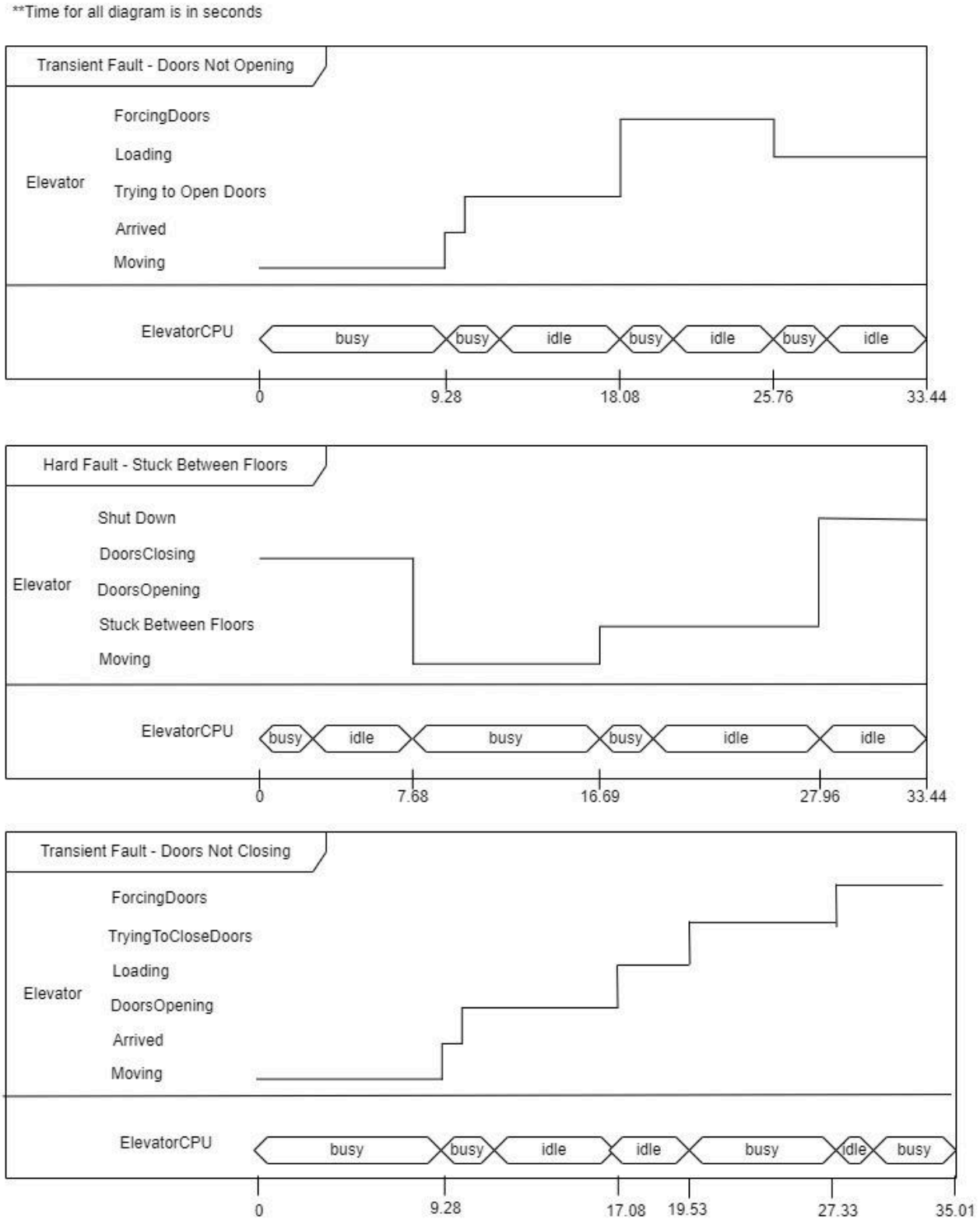
Figure 3: UML Sequence Diagram of the ElevatorSystem



2.4 Scheduler Timing Diagram

Figure 4 illustrates the Scheduler Timing Diagram for the ElevatorSystem.

Figure 4: Scheduler Timing Diagram for the ElevatorSystem



3.0 SETUP AND TEST INSTRUCTIONS

The ElevatorSystem can be cloned from the GitHub repository found at: https://github.com/nnicolell/elevator-system. IntelliJ and JDK version 21.0.1 was used to develop the ElevatorSystem.

To run the ElevatorSystem, run the main method found in ElevatorSystem.java in IntelliJ.

To run the ElevatorSystem tests, run ElevatorTest.java, FloorTest.java, HardwareDeviceTest.java, and SchedulerTest.java in IntelliJ.

4.0 MEASUREMENT RESULTS

The Canal Building elevators were used to measure the time it took to load and unload an elevator car, and move between adjacent floors.

These measurements were used to determine the timing of events in the ElevatorSystem.

Table 6: Summary of the Results for the Canal Building Elevator Moving from Floor 1 to Floor 7

Floor Splits	Load Time	Time Between Floors	Unload Time
1 to 2	6.42	8.44	10.64
2 to 3	9.41	9.02	7.72
3 to 4	6.84	10.02	7.66
4 to 5	6.38	9.39	6.86
5 to 6	6.61	9.61	6.35
6 to 7	6.49	9.18	6.85
Mean ($\overline{X}$)	7.03	9.28	7.68
Sample Variance (S^2)	1.39	0.29	2.38
Standard Deviation (S)	1.18	0.54	1.54

Table 7: Summary of Elevator Events at a 95% Confidence Interval

Elevator Event	Minimum Time Interval (s)	Maximum Time Interval (s)	Mean (s)
Load	5.79	8.26	7.03
Unload	6.06	9.29	7.68
Floor to Floor	8.71	9.84	9.28

Table 8: Load and Unload Times of an Elevator Car Based on the Number of Passengers

# of Passengers	Load Time (s)	Unload Time (s)
0	9.78	9.78
1	10.25	9.76
2	9.96	10.12
3	10.21	11.12
4	10.72	11.22
5	11.67	11.34

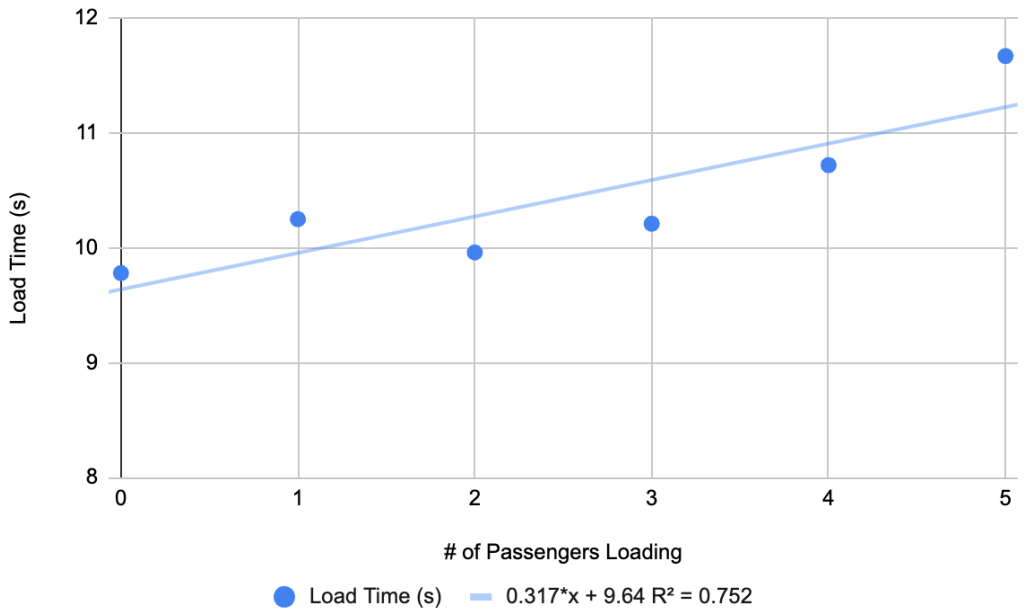


Figure 5: Load Times of an Elevator Car Based on the Number of Passengers

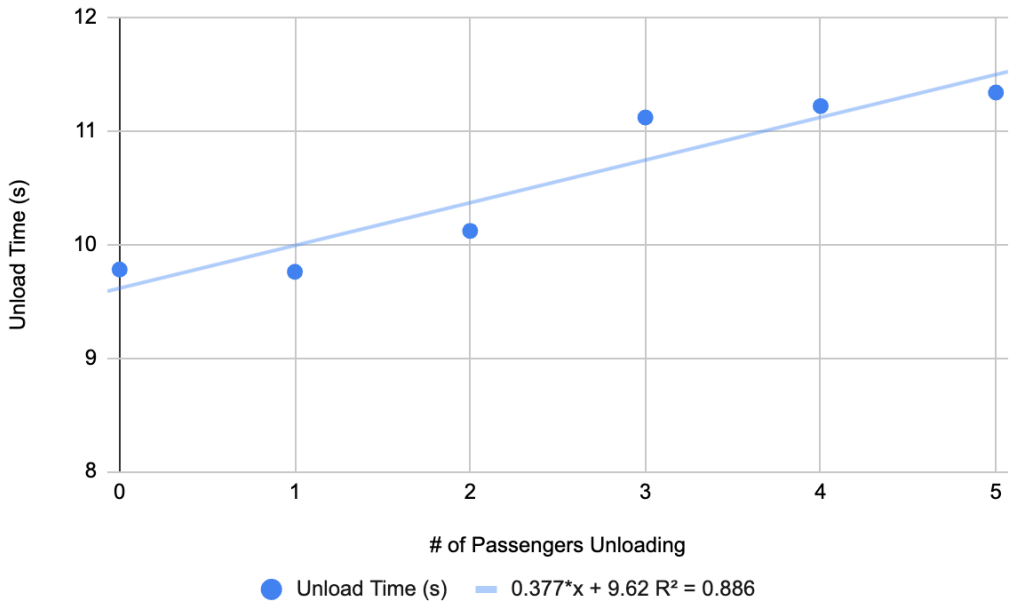


Figure 6: Unload Times of an Elevator Car Based on the Number of Passengers

5.0 REFLECTION

5.1 Positives

In the Elevator System project, our group did a good job at making our implementation dynamic. Therefore if certain attributes within the system, such as the number of elevators or the number of floors, needs to be changed, it requires minimal change to the code. For example, if 4 elevators are required, all that needs to change is ensuring there are 4 elevator ports and updating the user interface parameters in the ElevatorSystem main method.

Furthermore, the group's implementation of faults and correction was implemented in an efficient manner. There were timers to check if the doors were stuck open or closed, or if the elevator was stuck between floors. If the timer reached a certain threshold, such as 7.8 seconds, without the doors opening, the Elevator would acknowledge that there was a transient fault, and force the doors open. Additionally, if the timer continued for 11 seconds without the Elevator moving floors, the system would acknowledge the hard fault and shut down the elevator. Therefore, this implementation ensured that all types of fault were handled correctly and efficiently.

Finally, our group created tests that tested all aspects of our elevator system. The tests allowed us to identify bugs and logical and timing issues in the system. Our tests ensured that the system was working as expected and correctly.

5.2 Improvements

An improvement that could have been made in the elevator system is a different way for the Scheduler to decide which elevator gets a specific floor event. Currently there is one method that handles distributing event requests. If we were to refactor this part of the system, we would have had two separate methods, one to distribute events to elevators that are idle, and another one to distribute events to elevators that are currently running. This would allow for better organization for where events are going when.

Another improvement that could have been made is a different way for the elevators to run additional picked up events. For example, if an elevator was running an event to go from Floor 2 to Floor 9 and while it was running, the elevator also picked up an event to go from Floor 5 to Floor 7, currently, the elevator will completely the first event, going to Floor 9, and then complete the second event. If we were to refactor it, ideally the elevator would finish whichever event has the closest destination to its current location first. This would help the elevators run more efficiently.