# Final Report

Group 5:

Arman Kocharyan – 100888381

Katherine Nelson – 101012786

Andrew Dodge - 100938015

Moh Gahelrasoul – 101007118

Roman Kishinevsky – 101009733

Dr. Gregory Franks

SYSC 3303A – Real-Time Concurrent Systems

Wednesday, April 10th, 2019

# **Table of Contents**

[Title Page 1](#_Toc5559279)

[Table of Contents 2](#_Toc5559280)

[Breakdown of Responsibilities 3](#_Toc5559281)

Iteration 1…………………..……………………………………………………………………………3

Iteration 2…………………..………………………………………………………………..…….……3

Iteration 3…………………..…………………………………………………………………………...4

Iteration 4…………………..……………………………………………………………………………4

Iteration 5………..………………………………………………………………………………………4

[Diagrams 5](#_Toc5559282)

UML Class Diagram – Floor……………………………………………………………………………#

UML Class Diagram – Scheduler………………………………………………………………….……#

UML Class Diagram – Elevator………………………………………………………………………...#

State Machine Diagram – Scheduler……………………………………………………………………#

Sequence Diagram………………………………………………………………………………………#

Timing Diagram – Scheduler……………………………………………………………………………#

[Instructions 7](#_Toc5559283)

Set Up……………………..……………………………………………………………………………#

Testing……………………..………………………………………………………………..…….……#

[Measurement Results 8](#_Toc5559284)

[System Analysis 9](#_Toc5559285)

[Reflection 10](#_Toc5559286)

# **Breakdown of Responsibilities**

*Iteration 1*

|  |  |
| --- | --- |
| **Group Member** | **Responsibilities** |
| Arman Kocharyan | * Created the elevator subsystem |
| Katie Nelson | * Created the floor subsystem |
| Andrew Dodge | * Created the scheduler subsystem * UML class diagram |
| Mohamed Gahelrasoul | * Created the scheduler subsystem * State machine diagram for scheduler |
| Roman Kishinevsky | * Created the scheduler subsystem * State machine diagram for elevator |

*Iteration 2*

|  |  |
| --- | --- |
| **Group Member** | **Responsibilities** |
| Arman Kocharyan | * Test cases * Timing of elevator carts * Created cucumber tests and the logger * Elevator message (the default message class for all UDP messages) |
| Katie Nelson | * Floor subsystem arrival * Floor controller * Sensors * Implemented pickUpPerson, rideToFloor, and the event listener which opens a socket on a given port and listens until message is received |
| Andrew Dodge | * UML class diagram * Implemented startListen which starts a new thread/daemon that blocks and waits call |
| Mohamed Gahelrasoul | * Documentation * Timing diagrams * Worked on resetting the buttons, and elevator arrival |
| Roman Kishinevsky | * Requesting elevator up and down, with the up down buttons * Small interactions with the elevator |

*Iteration 3*

|  |  |
| --- | --- |
| **Group Member** | **Responsibilities** |
| Arman Kocharyan | * Worked on exceptions * Worked on testing |
| Katie Nelson | * Worked on exceptions * Worked on testing |
| Andrew Dodge | * Worked on timing * Worked on diagrams |
| Mohamed Gahelrasoul | * Worked on timing * Worked on diagrams |
| Roman Kishinevsky | * Worked on timing * Worked on diagrams |

*Iteration 4*

|  |  |
| --- | --- |
| **Group Member** | **Responsibilities** |
| Arman Kocharyan | * Got the scheduler working on a separate computer. * Calculated the mean and variance of the scheduler |
| Katie Nelson | * Worked on exceptions * Worked on testing |
| Andrew Dodge | * Got the scheduler working on a separate computer. * Calculated the mean and variance of the scheduler |
| Mohamed Gahelrasoul | * Worked on timing * Worked on diagrams |
| Roman Kishinevsky | * Worked on timing * Worked on diagrams |

*Iteration 5*

|  |  |
| --- | --- |
| **Group Member** | **Responsibilities** |
| Arman Kocharyan | * GUI |
| Katie Nelson | * GUI * Refactoring code to accept multiple people |
| Andrew Dodge | * GUI * Created, wrote, and formatted the report |
| Mohamed Gahelrasoul | * GUI * Created sequence diagram |
| Roman Kishinevsky | * GUI |

# **Diagrams**

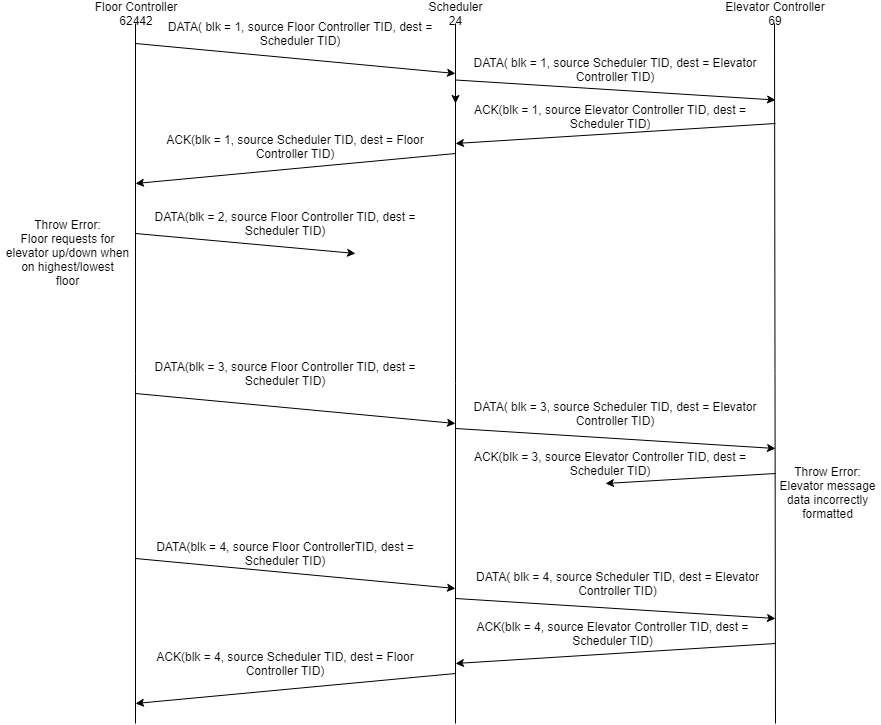
*UML Class Diagram – Floor*

*UML Class Diagram – Scheduler*

*UML Class Diagram – Elevator*

*State Machine Diagram – Scheduler*

*Sequence Diagram*

*Timing Diagram – Scheduler*

# **Instructions**

*Set Up*

To run the code:

* Open eclipse ide and import project as a maven project
* Open the subsystems directory

1) Right click elevator and run as java program

2) Right click scheduler and click run as a java program

3) Right click floor and click run as a java program

To run on multiple computers:

* Open eclipse ide and import project as a maven project on both machines
* Open the subsystems directory

1. In the core package run the IPGetter.java file on both machines
2. In the scheduler package open the Scheduler.java file on machine 1, set the public static final String ADDRESS = ""; to the IP address of machine 2, add it inside the ""
3. open the Floor.java file from floor package and Elevator.java file from the elevator package, and add machine 1's IP address to the public static final String ADDRESS = ""; variable in those files
4. Run Scheduler.java on machine 1
5. Run ElevatorController.java followed by FloorController.java on machine 2

*Testing*

* Open eclipse ide and import project as a maven project
* Open the subsystems directory

1. Run CucmonberRunner.java as a junit test

# **Measurement Results**

|  |  |  |  |
| --- | --- | --- | --- |
| Time Elevator Takes to Complete Trip in Seconds  (System.nanoTime was used to measure these values then was converted to seconds) | | | |
|  | Iteration 1 | Iteration 2 | Iteration 3 |
| Floor 2 to 3 | 3.018 | 3.016 | 3.020 |
| Floor 0 to 4 | 12.016 | 12.019 | 12.021 |
| Floor 3 to 2 | 3.017 | 3.016 | 3.018 |
| Floor 1 to 7 | 18.018 | 18.018 | 18.023 |

|  |  |  |
| --- | --- | --- |
| Time Elevator Takes to Move One Floor in Seconds | | |
| Iteration 1 | Iteration 2 | Iteration 3 |
| 3.018 | 3.016 | 3.020 |
| 3.004 | 3.004 | 3.005 |
| 3.017 | 3.016 | 3.018 |
| 3.003 | 3.003 | 3.003 |

*Total Mean:* 3.011s

*Total Variance*: 0.055s

# **System Analysis**

# **Reflection**

Our group quite enjoyed working on the project, we found it was an interesting experience. It allowed us to program and design a system using technologies that we had never had the skills to use before. As well as gave us the knowledge to attempt project like this in the future. When it comes to elements of the design that we enjoyed it mainly had to do with the code base as opposed to the GUI. We enjoyed how we were able to create a system that was compact and visually appealing for the grandeur of its features. Furthermore, we found that we created a system with code that was laid out nicely and was rather simple to follow if you knew base computer science knowledge. All in all, we were content with the code base however, it was the GUI and small extra elements that our group had an issue with in the end. The parts we believed should be redone if we had the time mostly relate to the GUI, there are some elements that do not stem from the graphics component of the project though they are minimal. Speaking of these minimal changes first, the main complaint we found came from us barely refactoring our code base. While it is true that it is an easy to follow code base, and simple if you understand that language we found that there are a number of redundant code elements within the system. With proper refactoring we would have been able to remove these code snippets and create a system we could be fully proud of. Furthermore, our only other code complaint stems from the fact that we barely documented any of our code, which is why it was stated that the code was easy to read if you had prior knowledge. The lack of comments makes it confusing for someone who is reading the information for the first time and given more time we would have liked to fully document the entirety of the codebase. Finally, our main complaint with the system, and the element we wish we could re do the most is the GUI. Due to outside factors we left the design of this portion of the system very near to the submission deadline. We still managed to finish everything on time, but it was not as polished as we would have liked it to be. We would have wanted to make it more fleshed out and showing the entire span of options that have to do with an elevator, as well as potentially show some amusing interactions with the passengers. To end, we enjoyed working on the project and creating this elevator system. We are quite proud of the code we have designed as a group; however, we all have decided that our visual skills need a serious overhaul.