SOFTWARE ENGINEERING

SOFTWARE REQUIREMENT SPECIFICATION

AUTOMATIC ELEVATOR CONTROL SYSTEM

Group Members :

Abhinandan (IIT2020119)

Shivam Harjani (IIT2020121)

Aditi (IIT2020138)

Kirti (IIT2020142)

Shashwat Mittal (IIT2020157)

1. Introduction

This document is the complete product requirement specification for an elevator controller software system for low-rise building elevator systems. Henceforth, this is the only document that contains all information regarding the requirements placed on this elevator controller by the stakeholders, cataloged in an unambiguous fashion. Unless otherwise stated, this document, and any future revisions of this document, supersedes all other requirements documents that exist for the said elevator controller system.

1.1 Purpose

We are required to develop a GUI enabled automatic elevator system in the CC3 building. The system functions with the automatic door opening and closing mechanism, and can also detect if the elevator is overloaded, where if a maximum capacity is met the door stays open and an alert message along with the sound is displayed on the interface and when people exit from the elevator then only the door will close. Later people can navigate on the floor and exit or enter with the same mechanism as discussed above. In case of failure an email or alert should be sent to the admin to make sure that lift reaches the nearest floor with some power backup and ensure the safety of the people.

1.2 Document Conventions

Deluminate : The opposite of illuminate. To turn off the light of something that is lighted.
EC Elevator Controller : A software system used to control and choreograph elevator hardware components (cars, sheave motors, doors, etc.) so that elevator passengers can be transported between floors of a building.
Elevator doors / doors: Refers to the inner door on an elevator cab and the corresponding outer door on the elevator shaft at the same floor at which the elevator cab is currently stopped at.

1.3 Product Scope

The overall objective of the Elevator System is to demonstrate the maximum passenger convenience and service within the constraints of available equipment.
The software of the elevator controller shows how responsible it is for the safe and efficient operation of all of the other components within the elevator system. The controller's main goal is shown as how to handle input signals from other components and respond accordingly to the output signals. Two of the main computational obligations of the controller are to have a

queuing system to log and process requests from passengers and to navigate the cabs of the elevators between floors in response to those requests.
It is important to note that this SRS document only pertains to the requirements of the software of the elevator controller. It does not include requirements for the hardware that it will be deployed on, just the software portrait of the same.

1.4 References

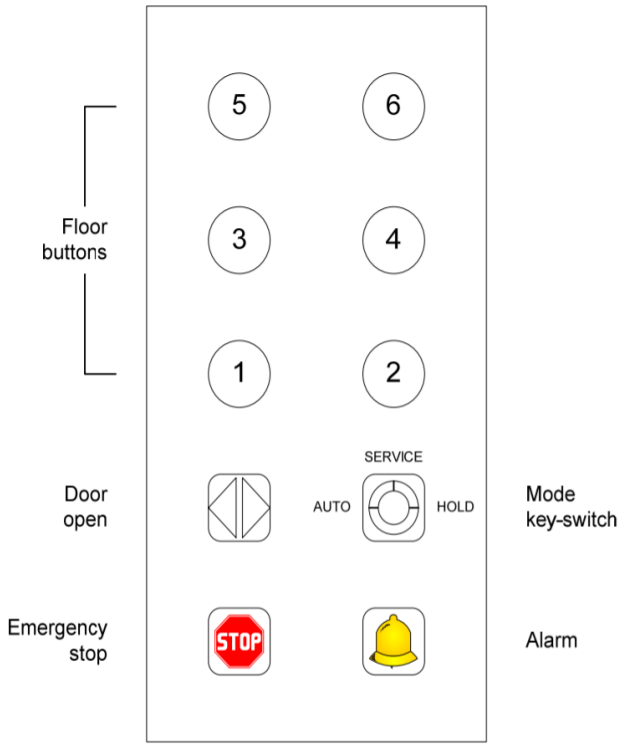
How does the elevator work ?
https://dev.to/jimjja/how-does-an-elevator-works-1g40
How to program an elevator ?
https://group.schindler.com/en/media/stories/how-do-you-program-a-modern-elevator.html
The Elevator Problem
https://web.eecs.umich.edu/~baveja/RLMasses/node2.html
Elevator – Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/Elevator

2. Overall Description

Low-rise buildings typically have three or more floors. To assist building residents and visitors in traversing these floors, simple two-car, cable-driven elevator systems are often installed. As with all modern elevators, an elevator controller (embedded software/firmware system) is required to control the movement and operation of the two elevators. This SRS document specifies a software design of elevator controller specifically for lowrise building elevator systems.

2.1 Perspective

Just like a typical elevator hardware building configuration is assumed, as shown below-



Our GUI also targets to provide the same to the users, having all the controls and visualization of the same.

2.2 Functioning

The major function of the EC is to show the control and the up and down movement of the elevator cars. It does so to position the elevator cars at floors where passengers have selected (either for pick-up or drop-off). It also depicts the opening and closing of elevator car doors and floor entrance doors to allow the safe entry and exit of passengers into and out of the elevator cars. In addition, the EC shows the control of illumination and delumination of floor indicators and buttons.

2.3 User Characteristics

The user should be familiar with the working of the elevator system as they are the elevator controller, essentially the other components of the elevator system that interact with it. The integral characteristic that these other components must have is being able to send and/or receive signals from the elevator control system. They must use the same protocol or format for signals as the controller does, so that communication between them is feasible.

2.4 Principal Actors

The principal actors here would be our user only, as they are the one using the software and there is no need for others.

2.5 Design and Implementation Constraints

1. The elevator software should work in such a manner showing proper closing and opening of the doors of elevator cars as well as the emergency calls.
2. General functionality of the overall elevator control system (not to be confused with EC subsystem) is already established and is fixed (e.g. the two operational modes, the recall mode, the expected sequence of illumination/delumination of buttons or indicators, etc.). Therefore, the EC must not redefine such generally understood/expected features or behaviors.

2.6 User Documentation

2.7 Assumptions and Dependencies

The following is the list of assumptions that we made about the automatic elevator control system -

1. There are 6 floors to which we will provide service.
2. It is supposed to display the movement of persons and elevator shafts.
3. There are a certain number of emergency functions throughout the system, and deals with them separately when invoked.
    a. Emergency bell pressed
        - Causes the bell to ring.
    b. Emergency stop button pressed
        - Causes the cab to stop and call the emergency services.
    c. Load sensor detector
        - Likewise, this only affects the cab with too much weight inside, it does not enter any specific mode of emergency operation, just gives a warning and halts the movement.
    d. Service switch
        - Switching a cab into hold mode can be performed manually in the event of an emergency, but not necessarily, it does not enter any specific mode of emergency operation.

3. External Interface Requirements

3.1 User Interface

There are no such specific requirements here because the elevator control system has no explicit user interface beyond the indicators and hardware specific requirements here.

3.2 Hardware Interface

Signals will be sent between the elevator controller device and other elevator components are assumed to be available and compatible with the controller device being programmed, as discussed in the above documentation. These logical events and function calls are listed in the table below for each elevator component.

Elevator Component	Function	Description
Elevator Controller	Turn on() Turn off()	Turn on the elevator system Turn off the elevator system
Floor Indicator	illuminate([in] floor)	Illuminate the floor number specified by floor (deluminate all other floor numbers).
Up/Down Button Panel	_enterEntranceRequest (sourceFloor, direction) deluminate([in] button)	Request entered (from elevator entrance) event Deluminate the button specified by button. Button is a value in {up, down}
Floor Door	open() close()	Open the elevator entrance floor door. Close the elevator entrance floor door.
In-Car Button Panel	changeMode(newMode)	This event is fired when the mode

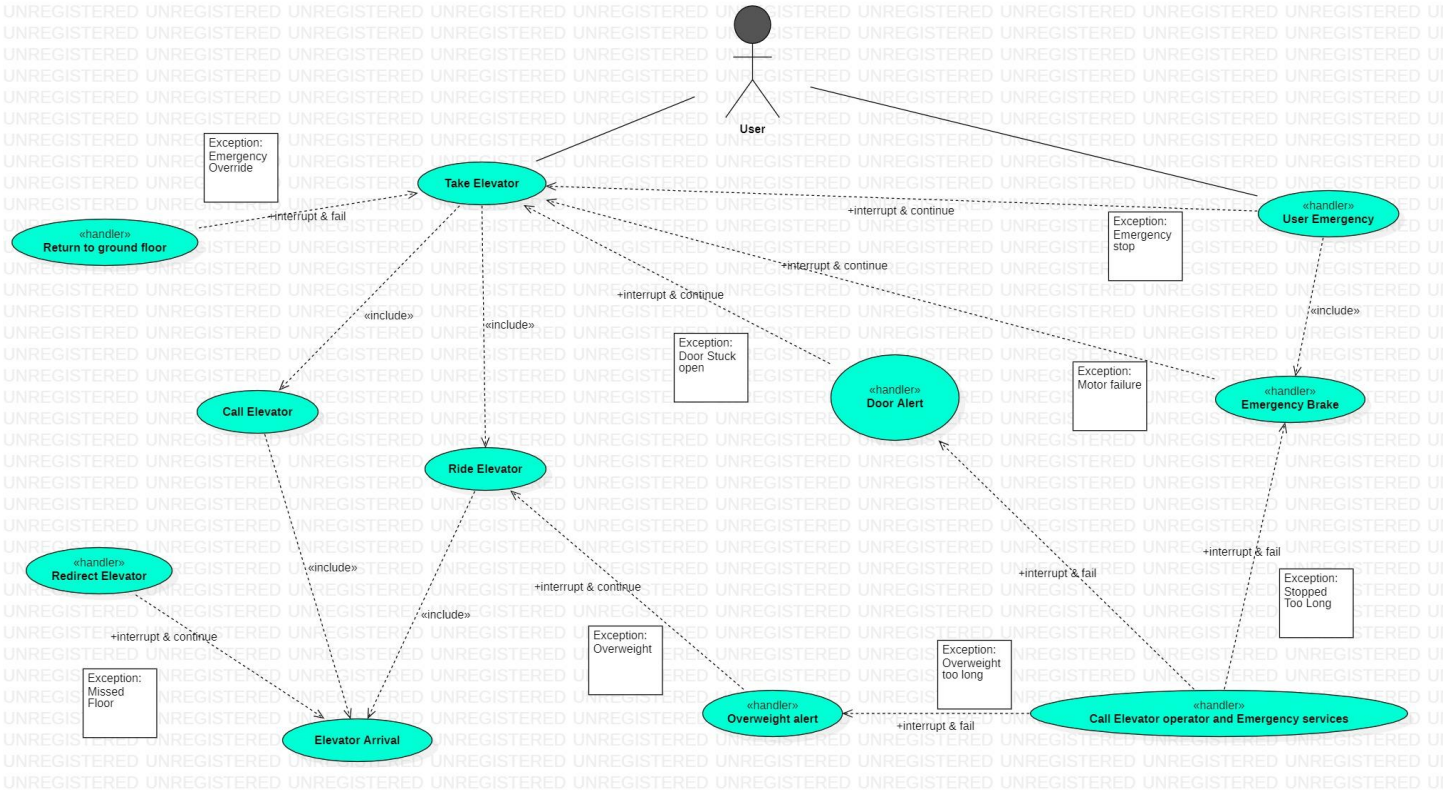
		key-switch is turned to a new position.
	_enterInCarRequest(request)	Request entered event, with request indicating the button being pressed (floor/action being requested).
	_doorOpenReleased()	Door open button released event.This event is fired whenever the door open button is released after being pressed (and held).
	_alarmReleased()	This event is fired whenever the alarm button is released after being pressed (and held).
	deluminate([in] button)	Deluminate the button specified by button.
In-Car Floor Indicator	illuminate([in] floor)	Illuminate the floor number specified by floor (deluminate all other floor numbers).
Load Sensor	_overLoad(bool)	Overload status change event, with new status indicated by bool. This event is fired whenever the elevator load changes from within load limits (bool is false) to above load limits, i.e. overloaded (bool is true), or vice versa.
Car Door	open()	Open the elevator car door.
	close()	Close the elevator car door.
Alarm Bell	toggleSound(bool)	Toggle the alarm bell as specified by bool (true for sounding, false for silenced). bool is a value in {true, false}

3.3 Communication Interface

As in this, we are just going to have direct signaling between the components and emergency calls in situations so there won't be any specific requirements for the same.

4. Functional Requirements

The sections that follow formally specify the functionality of our elevator control system as well as the various interface requirements. The use case diagrams provide functional requirements. The following use case diagram displays the requirements and the flow of the control system and also groups use cases into related concerns.



4.1 Overall Elevator System

The overall elevator system is the high-level construct with which human users, passengers and operators, interface.

4.2 Turn On Button Handler

Turn on the elevator system, invoke the Elevator start-up sequence.

4.2 Turn Off Button Handler

Turn off the elevator system, invoke the shut-down sequence.

4.3 Emergency Button Handler

In case of failure, send an email or alert to the admin to make sure that lift reaches at the nearest floor with some power backup and ensure the safety of the people.

4.4 Overweight Alert Handler

In case the elevator is overloaded, where if a maximum capacity is met the door stays open and an alert message along with the sound is displayed on the interface and when people exit from the elevator then only the door will close.

5. Other non functional requirements

This SRS only pertains to the requirements of the software of an elevator controller, not the hardware that it runs on. Therefore, the details of hardware interface requirements are out of the scope of this document. In order for the software to work properly, the elevator controller hardware must provide a way of transmitting and receiving communication signals from other components of the elevator system. They are simply triggered or handled by the controller software depending on whether the signal is input or output to the controller.
Some the major requirements that we can have in the same are:

- For the safety of the passengers inside, the inner doors of a lift should not be opened unless the lift is stopped at a correct floor position – they should never open while a lift is moving.
- For the safety of potential passengers, the outer doors of a lift should not be opened at a particular floor unless there is a lift stopped at that floor in that shaft.
- For the safety of the passengers inside, a lift should not move until its doors are completely closed.
- The order in which a lift answers its active floor requests minimizes the number of changes in direction the cab has to perform.
- In case of an emergency the moving lift stops at the nearest floor in the moving direction, and also issues an emergency call