Elevator Controller Case-Study

1. Introduction

This document is inspired by a real Software Requirement Specification (SRS) of an **Elevator Controller (EC)** software system for low-rise building elevator systems. It describes, in a simple way, some requirements and assumptions placed on this elevator controller by the stakeholders. The intended audience for this SRS is software engineers implementing the specified elevator controller and stakeholders of either the elevator controller itself or the elevator system/building where the controller will be deployed.

An elevator controller controls and choreographs elevator hardware components (cars, sheave motors, doors, etc.) so that elevator passengers can be transported between floors of a building. An elevator controller must interface with various input/output components that interact with elevator users (passengers and operators).

This document covers the details of the elevator controller system (the system). Section 2 describes the external systems and/or environments in which the elevator controller system shall work, with enough detail to complete an implementation of the same. Section 3 describes the software architecture and functional requirements of the elevator controller.

2. General 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 an elevator controller specifically for two-car low-rise building elevator systems.

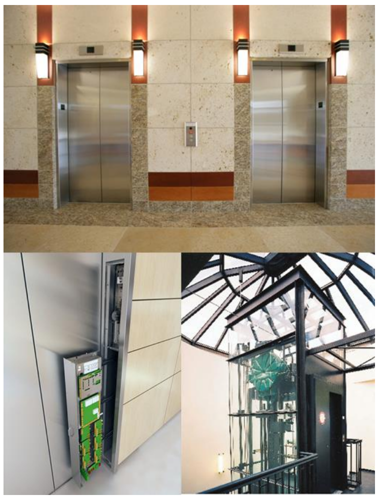

Figure 1. Elevator system overview: floor doors (top), controller (bottom left), motor (bottom right)

Figure 1 shows an example of an elevator system for low-rise buildings.

2.1 Product Perspective

Typical elevator hardware (shafts, cars, cables, floor doors, etc.) and hardware-building configuration is assumed. The following figures highlight (example) components visible to typical elevator end-users (passengers and operators). Some of these components will interface with the elevator controller. A six-floor low-rise building is used for example.

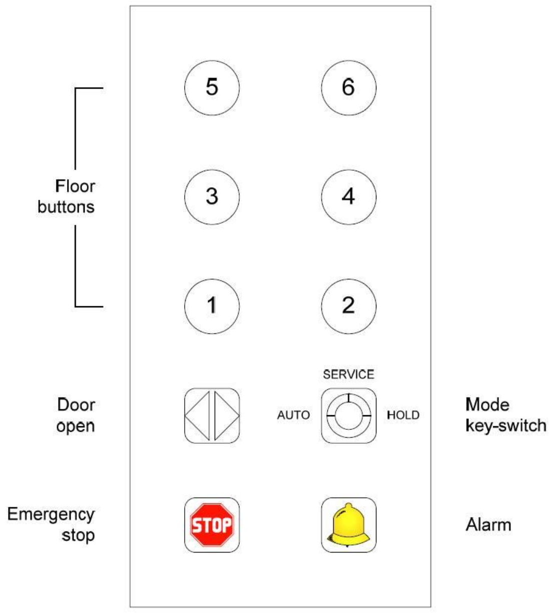

Figure 2. Example in-car button panel

Figure 2 shows a typical button panel inside each elevator car. The button panel has the following buttons/switches: floor buttons 1 to 6, door open button, emergency stop button, alarm button, and elevator mode key-switch (AUTO/SERVICE/HOLD). Elevator modes, emergency stop behavior, and alarm behavior are detailed in subsequent sections.

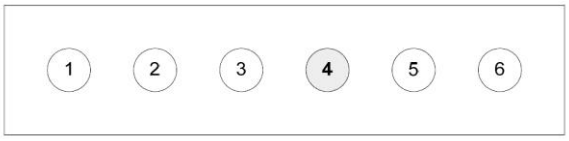

Figure 3. Example floor indicator

Figure 3 shows a typical floor indicator panel installed above each elevator door on each floor of the building, and above the door inside each elevator car. The display panel illuminates the number corresponding to the current position (floor) of the elevator car that it represents.

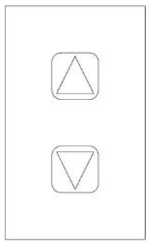

Figure 4. Up/down button panel

Figure 4 shows a typical up/down button panel installed at each elevator entrance located on each floor of the building. The up/down button panel is used by passengers awaiting pick-up to indicate their desired direction of travel. Each elevator entrance has two elevator doors, one for each elevator car, on either side of the up/down button panel. There is only one up/down button panel per elevator entrance.

The purpose of the elevator controller is to control elevator hardware components (cars, sheave motors, doors, indicators, etc.). The elevator controller is a software system that is (presumably) flash-loaded as firmware on a hardware controller device located in the building's elevator control room. Once running, the EC accepts input signals sent to the controller device from other elevator components, and outputs signals to elevator components to control their behavior. The EC is a sub-system of the overall elevator system (the high-level construct with which human users, passengers and operators, interface).

2.2 Product Functions

The primary function of the EC is to control the up and down movement of the elevator cars (by way of controlling the elevator sheave motors). It does so to position the elevator cars at floors where passengers have selected (either for pick-up or drop-off). It also controls 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 controls illumination and delumination of floor indicators and buttons. The EC supports two operational modes and one recall mode: AUTO, HOLD, and SERVICE. The specific mode in which the EC is operating determines the way in which it controls the up-down movement of the cars and open-close actions of the doors. The mode is selected per elevator car using the mode key-switch on in-car button panels. Behavior of each mode is as follows:

- In **AUTO** mode (operational), the elevator behaves as a typical elevator would. Elevator cars are sent to floors where pick-up or drop-off requests have been made (via up/down button panels at each elevator entrance or in-car button panels, respectively)
- In **HOLD** mode (operational), the elevator behaves as a service elevator, ignoring any passenger pick-up requests, but instead changing floors only when floor selection is made by a passenger inside the elevator car (using the in-car button panel). While in HOLD mode, the elevator doors stay open indefinitely at each destination floor (doors are closed while the elevator is in motion, of course). Only one destination floor can be selected at a time while in this mode.

HOLD mode is generally used to facilitate move-ins (i.e. someone moving into a building) or planned transportation of large items.

- In **SERVICE** mode (recall), the elevator is returned to a (pre-configured) default/recall floor and remains on that floor with the doors open. Reanimation of the stopped elevator car then requires operator action (operator must use the in-car mode key switch to change to one of the two operational modes).

2.3 User Characteristics

There are two types of eventual end-users of the overall elevator system: passengers and operators.

A **passenger** is anyone who wishes to enter an elevator car on floor i, select a destination floor j (1 <= i, j <= # of floors, i != j), "ride" the elevator car until it arrives at floor j, and exit the elevator car. A passenger is expected to be able use up/down buttons to request elevator pick-up, to enter and exit an elevator car through elevator doors, and to select desired destination floor(s) by pressing buttons on in-car button panels.

An **operator** is someone who performs any of the following:

- Changes the elevator mode (AUTO/ HOLD/SERVICE) of an elevator car using the key-switch on in-car button panels.
- Turns the elevator system ON and OFF using a key-switch in the elevator control room.

An operator is expected to know how to use a key-switch (given the correct key) and understand the consequences of his configuration choices (made via key-switch). An operator is also a passenger.

2.4 Assumptions

The EC specified in this SRS is done so *specifically* for low-rise building elevator systems. In a low-rise setting, the EC need not implement any complex optimization techniques for scheduling car movement. Car selection for passenger pick-up will simply be determined by which car is closest for passenger pick-up (and if already traveling, is going in the logically correct direction), and cars will simply wait at the last-stopped floor for more passenger requests (as opposed to, say, going to a floor that is statistically most likely to have the next passenger request). There will not be any logging of usage statistics.

The following assumptions are made regarding the *desired/standard* behavior of low-rise building elevator systems:

- Each elevator car is equipped with an **alarm bell**. When the alarm button of an in-car button panel is *pressed-and-held*, the alarm bell for that car is set to sound and continues to sound until the button is released. While the alarm bell is sounding, the elevator will continue operating as usual. Therefore, the purpose of the alarm is merely to notify elevator operators that attention is required (for whatever reason) – it does *not* imply that the elevator must be halted.

- The **emergency stop button** of an in-car button panel is intended to give passengers the option of returning the elevator car to a default recall floor immediately in case of emergency. When this button is pressed, it is assumed that the elevator car is to go to a (pre-configured) default/recall floor and open the elevator doors. Reanimation of the stopped elevator car then requires operator action (operator must use the in-car mode key switch to reset the stopped car or restart the entire elevator system). Notice that pressing the emergency stop button is functionally equivalent to changing the elevator to the recall SERVICE mode (using the mode key-switch).
- It is assumed that a building's **Fire Detection System** does have some control over the elevator system in the event that a fire is detected. For simplicity, it is assumed that the Fire Detection System simply triggers an elevator system shut-down in the event of a fire.
- It is assumed that the elevators are to be returned to a (pre-configured) default/recall floor upon system shut-down, and left on that floor with all elevator doors open. For simplicity, it is also assumed that elevators are to start on the same default/recall floor upon system start-up (elevator cars will be returned to the default/recall floor as part of the start-up sequence in case they were somehow manually relocated while in the OFF state).

A single EC instance controls both elevators cars and related components.