Software Requirements Specification

for

Functional Elevator Simulator

Version 0.1.0

Luis Alejandro Barreda Acevedo 2020425927
Rubén Isaí Hurtado Granados 2023282072
Josué Daniel Soto González 2023207915
Samuel Zúñiga Vega 2023029693

Tecnológico de Costa Rica

20/02/2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Initial documentation	20/02/24	Initial information on the project filled with requirements.	0.0.1
Update of requirements	22/02/24	Updated the information based on consultation. More information added on the base prototype and requirements.	0.1.0

1. Introduction

1.1 Purpose

The purpose of this Systems Requirements Specifications (SRS) document is to provide the guidelines necessary to design and implement the Functional Elevator Simulator Software (FESS). Therefore, this document could be used by any member of the development team to build a software application that satisfies the requirements stated by the client.

The SRS describes the components and functionalities that the FESS is expected to accomplish. As well as the Graphical User Interface (GUI) design which meets the use cases proposed by the client. It also includes nonfunctional requirements that can be used as a final criterion to judge if the software fulfills the SRS.

1.2 Document Conventions

The following typographical conventions and standards were applied in the making of this SRS. Titles are written with the font Times, size 14 and bold. Normal text is written with the font Times, size 12. To highlight important aspects of the document the same convention in normal text is used but bold.

List of acronyms used in this document:

SRS – Systems Requirements Software, this document which outlines the requirements that the software must fulfill.

GUI – Graphical User Interface

UML – Unified Modeling Language.

FESS – Functional Elevator Simulator Software

EIR – External Interface Requirements

1.3 Intended Audience and Reading Suggestions

This SRS document is designed to cater to various entities involved in the development of the FESS. Considering the different roles that the involved parties play and the different interests in the project of each one, a set of guidelines are presented to outline the intended audience and some reading recommendations for each group. It starts with an Overall Description of the project followed by sections explaining the required features of the system in more detail as well as development guidelines and interface design. Appendices are also included to give additional information in case it is needed.

The following outlines the intended audiences and provides reading suggestions for each group:

Clients and Sponsors – The main interest of this group is to comprehend the overall vision, goals and, expected outcomes of the FESS, as such it is recommended to start by reading the Functional Requirements of the project to understand the main features and functionalities of

the software. Following this would be the Non-functional Requirements so that the readers can have a better grasp of the performance, safety, and security of the system. And lastly reading the Appendix is highly advised for any additional technical details concerning the project.

Developers – This group's main goal is to comprehend the structure of the software, because of this a series of specifications and implementation guidelines can be found in this SRS. The first step for a developer is to read the Project Scope to have a better understating of the end goal of the project, including objectives and benefits that the software must contain. Next up would be reading the Overall Description of the project as this section specifies the many features to be developed and sets up the working guidelines that must be followed when implementing said features. After reading this, the next crucial step is to read the External Interface Requirements as this outline the way in which the software interacts with the interface and relays the information to the user and vice versa. And lastly reading the Appendix is highly advised for any additional technical details concerning the project.

Users of the system – Their goal is to comprehend how to utilize the FESS effectively, to accomplish this a User Manual may be provided to give an overview of the system's functionalities and interactions. Additionally, the user may read the EIR to look at some of the expected inputs and/or outputs of the system.

1.4 Project Scope

The FESS aims to be a comprehensive simulation tool designed to replicate the operating behavior of elevators in the real world on multiple environments. Its primary purpose is to provide a better understanding of the intricacies that go into developing a functional elevator system, that and to comply with the instructions given by the professor of this course.
The main goal is to be able to simulate different case scenarios by modifying variable such as load capacity, height of the buildings and environmental hazards like earthquakes and similar, among others. This with the objective of obtaining a deeper understanding of the elevator systems which can then be used to improve the current models. With this in mind, the FESS also aligns with some more intricate corporate goals as it promotes innovation, improvement of operational efficiency and maintenance of a competitive edge in the industry.

1.5 References

D. L. Pepyne & C. G. Cassandras, "Design and implementation of an adaptive dispatching controller for elevator systems during uppeak traffic," in IEEE Transactions on Control Systems Technology, vol. 6, no. 5, pp. 635-650, Sept. 1998, doi: 10.1109/87.709499.

A. Arrieta, M. Otaegi, L. Han, G. Sagardui, S. Ali & M. Arratibel, "Automating Test Oracle Generation in DevOps for Industrial Elevators," 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, 2022, pp. 284-288, doi: 10.1109/SANER53432.2022.00044.

ASME A17.1 (2000). On cars with two entrances, a separate door-close button shall be provided for each entrance if both entrances can be opened at the same landing. Safety Code for Elevators and Escalators.

ACE Lifts (s.f.). Traction Lifts: an infographic on how they work. ACE Lifts. Archived from the original on 19 July 2014. Retrieved from https://web.archive.org/web/20140719082011/http://acelifts.com/news/traction-lifts-work/

2. Overall Description

2.1 Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

2.2 Product Features

<Summarize the major features the product contains or the significant functions that it performs or lets the user perform. Details will be provided in Section 3, so only a high level summary is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or a class diagram, is often effective.>

2.3 User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the favored user classes from those who are less important to satisfy.>

2.4 Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

2.5 Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

2.6 User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

2.7 Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

3. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

3.1 System Feature 1

<Don't really say "System Feature 1." State the feature name in just a few words.>

3.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

3.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

3.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:
REQ-2:

3.2 System Feature 2 (and so on)

4. External Interface Requirements

4.1 User Interfaces

A typical modern passenger elevator for a building can have multiple components for different functions. The hardware can vary for each context so the simulation will take in account the most important aspects. The required components that are relevant to the problem of the elevator in this software are the following:

- *Buttons outside the elevator: Buttons to make a call from any floor and the specified direction (signaled by an arrow or triangle). Top and bottoms floors only have a down and up button respectively. They should send signals to the elevator for it to attend to.*
- *Empty space inside: Space to stay as the elevator moves. This is where passengers must stay while it is on normal functioning.*
- *Overload sensor: Prevents the elevator from moving until the excess load is removed.*
- *Air conditioning: System that allows air to get inside and outside the elevator.*
- *Control Panel: A panel with buttons for the distinct functions of the elevator. The buttons are detailed in a separate list below.*
- *Sets of doors: Doors to keep the passengers in or out while the elevator is moving.*
- *Door sensors: Sensors that allow the door to detect if there is nothing that could obstruct closing. They should detect moving objects and static objects on the door's way.*
- *Telephone: Device that can allow a trapped passenger to call for help if an accident occurs. It should be connected to a local emergency system or the security system of the building.*
- *Restricted panel: A panel which can only be opened with a key or code that can allow the technicians to repair or see the state of the elevator.*
- *Walls: Walls that restrict the passengers from leaving or entering by unexpected ways.*
- *Speaker: A speaker to make defined sounds when entering or leaving a floor or to notify the elevator state. It can also be used to reproduce alarms in case of emergency.*

The elevator as addressed in the control panel should have buttons that make it move to the passengers desired destination. The panel should be inside and be the only way that the passengers must interact with the elevator. The buttons on the panel are listed and detailed as follows:

- *Floor number button: There should be a button for each floor on the building for the passengers to be able to choose their desired destination. It should send a signal for the elevator to move if the current direction matches with it.*
- *Emergency or call button: A button should be present to report any anomalies by a trapped passenger. Some can include a telephone that directs them to a central security system of the building.*
- *Door open and close buttons: There should be buttons for the passengers to manually open or close the doors when the elevator is at a floor. The buttons should make a call for the doors to close or open accordingly to the button pressed. A door should not be closed if the elevator detects an object or passenger obstructing its way. The elevator must not close when it has just opened at a floor even if the button for closing is pressed.*
- *Screen: A screen should detail the current floor and the direction the elevator is moving to. It can display other messages in case of anomalies or maintenance.*

The user interface includes a screen where the elevator is displayed and the n-number of users that are accessing or already accessed the elevator. Simultaneously, the interface provides three additional layouts, the external control pad from the elevator in which the up and down button will be enabled if the level is neither the highest nor the lowest; in case it is, it'll only show the down button for the highest level and the up button for the lowest one. Next, the internal control pad where a screen will show the current level and where is the elevator going after that (up or down arrow), the level buttons, the close or open door, the emergency button, and two speakers for illustration only. Finally, the master-like control pad to control the environment in which the elevator will perform its tasks (add a person to the elevator, subtract one, or cause an outage to simulate an actual emergency. For a better understanding of the user interface see Figure 4.1.1.

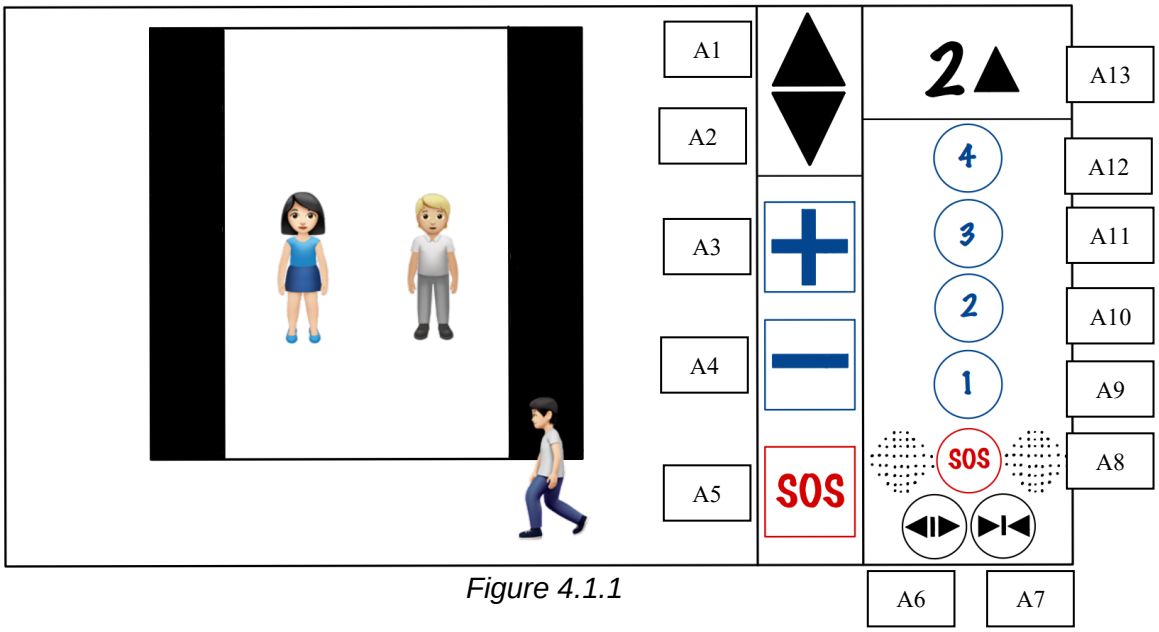

Figure 4.1.1

4.1.1.1

Item	Request Elevator Up
Purpose	Requests elevator letting it know the user wants to go up
Input	Mouse click
Output	Elevator goes to user's floor
Validity	Click within the button
Error Handling	If the elevator already passed the level, it should end the current cycle and then when going up/down again, stop at that level
Relationships	None
Reference	Figure 4.1.1, item A1

4.1.1.2

Item	Request Elevator Down
Purpose	Requests elevator letting it know the user wants to go down
Input	Mouse click
Output	Elevator goes to user's floor
Validity	Click within the button
Error Handling	If the elevator already passed the level, it should end the current cycle and then when going up/down again, stop at that level
Relationships	None
Reference	Figure 4.1.1, item A2

4.1.1.3

Item	Add user to elevator
Purpose	Add one more person to the elevator
Input	Mouse click
Output	One more user on the elevator
Validity	Click within the button
Error Handling	If the weight is too much, the elevator shouldn't work
Relationships	None
Reference	Figure 4.1.1, item A3

4.1.1.4

Item	Subtract user from elevator
Purpose	Subtract one user from the elevator
Input	Mouse click
Output	One of all current users gets down from the elevator
Validity	Click within the button
Error Handling	If there´s no more users, the button shouldn't work
Relationships	None
Reference	Figure 4.1.1, item A4

4.1.1.5

Item	Outage button
Purpose	Simulate an emergency scenario
Input	Mouse click
Output	An earthquake is generated in the screen
Validity	Click within the button
Error Handling	There should be no error
Relationships	None
Reference	Figure 4.1.1, item A5

4.1.1.6

Item	Open elevator
Purpose	Opens elevator's doors
Input	Mouse click
Output	The doors begin to open
Validity	Click within the button
Error Handling	If there was a process beginning to start, but the button is triggered, the action stops only if the elevator is at a floor level, if not, the process continues
Relationships	None
Reference	Figure 4.1.1, item A6

4.1.1.7

Item	Close elevator
Purpose	Closes elevator's doors
Input	Mouse click
Output	The doors begin to close
Validity	Click within the button
Error Handling	If there was a person entering the elevator, the doors should stop and reopen
Relationships	None
Reference	Figure 4.1.1, item A7

4.1.1.8

Item	Emergency button
Purpose	Calls the maintenance of the building because the elevator isn´t working as it should
Input	Mouse click
Output	The speakers begin to sound as if they were calling someone in charge
Validity	Click within the button
Error Handling	There should be no error
Relationships	None
Reference	Figure 4.1.1, item A8

4.1.1.9

Item	Select first floor
Purpose	Command the elevator to go to the first floor
Input	Mouse click
Output	The elevator moves to the first floor
Validity	Click within the button
Error Handling	If there were previous commands, the elevator should check and put the floor in order with the previous commands, if not, it can restart the cycle going to that floor directly.
Relationships	None
Reference	Figure 4.1.1, item A9

4.1.1.10

Item	Select second floor
Purpose	Command the elevator to go to the second floor
Input	Mouse click
Output	The elevator moves to the second floor
Validity	Click within the button
Error Handling	If there were previous commands, the elevator should check and put the floor in order with the previous commands, if not, it can restart the cycle going to that floor directly.
Relationships	None
Reference	Figure 4.1.1, item A10

4.1.1.11

Item	Select third floor
Purpose	Command the elevator to go to the third floor
Input	Mouse click
Output	The elevator moves to the third floor
Validity	Click within the button
Error Handling	If there were previous commands, the elevator should check and put the floor in order with the previous commands, if not, it can restart the cycle going to that floor directly.
Relationships	None
Reference	Figure 4.1.1, item A11

4.1.1.12

Item	Select fourth floor
Purpose	Command the elevator to go to the fourth floor
Input	Mouse click
Output	The elevator moves to the fourth floor
Validity	Click within the button
Error Handling	If there were previous commands, the elevator should check and put the floor in order with the previous commands, if not, it can restart the cycle going to that floor directly.
Relationships	None
Reference	Figure 4.1.1, item A12

4.1.1.13

Item	Screen
Purpose	Show the user information about the elevator's cycle (shows the current floor represented with a number and the current orientation represented with an up/down arrow). If there's an outage the screen should say so (assuming it has electricity after the outage) and if the A8 button is pressed, the screen should say (calling assistance)
Input	(Not by the user) Current floor from the cycle and current orientation
Output	Floor and orientation, outage or assistance messages
Validity	N/A
Error Handling	There should be no error
Relationships	None
Reference	Figure 4.1.1, item A13

4.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

4.3 Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

4.4 Communications Interfaces

The FEES does not require any form of communication via internet, email, etc. It is not connected to a database to save and load information because that function is performed using locally saved files. All communication with the software is made through the GUI, including important warning and emergency messages.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

5.2 Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: Issues List

< This is a dynamic list of the open requirements issues that remain to be resolved, including TBDs, pending decisions, information that is needed, conflicts awaiting resolution, and the like.>