Elevator Control System

Software Architecture Design

SAD Version 2.0

Team #3
8 October 2019

CS 460 Software Engineering

Contents

1 Introduction

Good software is identified by the end user for its features and functionality, by the client for its profitability and maintainability, and by the programmer for its legibility and clarity. It should be clear that all three pillars ultimately characterize good software, but more importantly that the three are interdependent. The developer must then guarantee all of the above for the sake of all entities involved. A top down approach has been taken up to this point. The feasibility study asked answered the fundamental question: Can it be done? The requirements definition document was then passed the baton and answered the next logical question: Within what parameters can it be done? The software specification document then answered: What is the desired behavior of the system? Now we may answer: How will ensure the desired behavior?

The ultimate goal is to ensure an efficient, safe, and maintainable implementation of the Cretaceous Gardens Controller (CGC) software[1]. To that end, all objects are illustrated in their proper contexts, and their crucial functions have been delineated as clearly as possible while simultaneously allowing the programmer enough flexibility so as to not stifle his or her creative process.

This is a road map for the eventual implementation of the system. It details the most relevant objects and their relationships in the form of diagrams. Explanations accompany all diagrams for the sake of clarity.

2 Design Overview

The class architecture presented here [2] *aims to maximize efficiency, maintainability, and safety. Without an efficient system, its safety may be compromised due to unnecessary delays between components. Maintainability can impact a safe implementation of the system if the system is permeable to programmer errors. The decoupling of concerns and a solid hierarchy are paramount. The design has been color-coded to increase readability. The colors are only for the sake of distinguishing one component from another.*

[1] Introduction by Zeke.

[2] Diagrams by Siri, Anas, Santi, and Zeke.

The component specification diagrams in the following section inherit these colors. Small red arrows point to objects that may be triggered by an event. Said objects are virtual devices and iconifications of their physical triggers have been connected to them with bidirectional blue arrows.

3 Component Specifications

Here are the class specifications [3] *for objects found in Section 2. The most important attributes and functions are given and, where appropriate, any expected preconditions, parameters, return values, and post conditions are included. Explanations are given for each component, but it should be noted that helper functions are not shown, as they are entrusted with the programmer. Important fields and helper fields are treated analogously.*

Some Class Name	
Attributes	
attribute 1	brief description
attribute 2	brief description
.	brief description
.	brief description
.	brief description
Functions	
function1()	→ brief description
function2()	→ brief description
.	→ brief description
.	→ brief description
.	→ brief description

[3] Component specifications by Anas and Siri

4 Sample Use Cases

A broad overview of use cases begins this section and it is followed by detailed case descriptions. Human actors are denoted by small stick figures and have the same color scheme as the box that contains their labels. The section later features use case diagrams for each actor and a detailed sample of use cases.

5 Design Constraints

Due to the real-time nature of the system, there exist some additional constraints[4]*. Namely, it must be the case that all data structures concerning the safety controls are as fast as possible but also that they are capable of prioritizing all signals in the best way possible.*

5.1 Safety

The safety is highly prioritized in our design of the ECS. We have considered associating the fire alarm directly with the ECS because in the case of the fire alarm triggering, this event has a very high priority and the ECS should immediately react to this event. When it comes to elevator and bay safety, each elevator and bay will have set of Safety Controls that they can communicate through and the ECS monitors the situation from the top. By ensuring safety for both bays and the elevators, the ECS operation will be carried smoothly without hurting the passengers.

5.2 Implementation Guidelines

According to the design, we suggest programmers to use some sort of Concurrent safe Messaging Queue for the communication between sensing objects and their associated parent objects. We also recommend using concurrent safe Priority Queue for the ECS, so the ECS can react based on the certain given priority. The priority should be considered because in the case of an emergency, the priority for that event should be at the very top so that the ECS should immediately react to it by closing its doors and going to first floor if its not there already.

[4]Design Constraints by Anas.

6 Definition of Terms

The following is a list of definitions contain the most commonly used technical terms within this document, whose meaning may not be immediately apparent to the lay reader. Most definitions are defined by the authors for use within the context of this document. Some may originate from vocabulary shared across the general references cited . In the event that a definition was taken directly from a source, it is followed by a citation. [5]

CGC: Acronym for Cretaceous Gardens Controller

DVR: Acronym for Digital Video Recorder

Electrical Conduction: The movement of electrically charged particles through a transmission medium.

GPS: Global Positioning System

Hardwired Ethernet: This references the latest IEEE standard for Ethernet utilizing physical cables.

Network: All nodes with which the CGC interacts, the links that connect them to each other and to the CGC, the CGC itself, and all related databases.

Node: The generic term that refers to any device connected to the CGC in any way. This includes autonomous vehicles, tokens, the T.Rex monitor, all electric fence panels, all kiosks, and all cameras.

Safely Inactive: A state in which a vehicle is fully functional and ready to be dispatched.

Safely Occupied: A state in which a vehicle contains at least one person, is locked, and is ready to depart.

Token: An interactive device used by the visitor that grants access to locations.

[5] This list is mostly a reduction of the term list found in the preceding Software Design Specification document.