Designing Software for Elevator system

Aditya Karmarkar[1], Mingling Chen[1], Liu Zheyuan[1]

[1]*Department of Computer Science, University of Delaware, Newark, USA*

Abstract

Purpose: Elevator system is broken down into several components and sub-modules, commonly know as Modularization of Software. Purpose of the paper is to mention and describe in brief architecture of elevator system using Comprises relation within module and Design of software system using Uses relation among mentioned module.
Methods: Identify Design components and their relations into system architecture
Results: Design and Architecture of Elevator system
Conclusion: Elevator system can be modularized leading to simple and much more understandable and readable design.

Keywords: Software Engineering, Design Document, Architecture of Software

1. Architecture of Elevator System

Every Software system can be modularized, keeping this general principal in mind , we'll try to modularize an elevator system. This document tries to capture all possible components, if anything is missing, then that will be included as part of Future work.

Modularization of Elevator system

1. Visualization
    - UI.
    - Simulator.
        - Start
        - End

Correspondence: Department of Computer Science, University of Delaware, Newark, USA.
E-mail: karma@udel.edu

2. Scheduler
    - RequestMadeFromFloor
    - RequestMadeInsideUnit
3. Operations
    - Movement.
        - Move
            * Up
            * Down
        - Stop
    - Indicator
        - Counter
        - Display
        - Buttons
    - Door
        - Open
        - Close
4. ElevatorUnit
    - Lights.
    - Motor.
    - Door.

Every Module hides Design decision in it, We'll try to give a brief idea behind every module and every module will be used in simulation of an Elevator software system.

- Visualization - This Component will create GUI of Elevator system where the client can see visual impact of every operation without knowing how system performs the operations, which will be hidden inside the Simulation stack, this submodule will use all underlying modules. This component has submodules, Simulation stack which operate and access hardware components through operation and Elevator Unit.
- Scheduler - This component is responsible for the arrival of an elevator unit when requested by a client or when a client is inside the elevator system. This component hides how elevator scheduling works, i.e. logic behind scheduling.

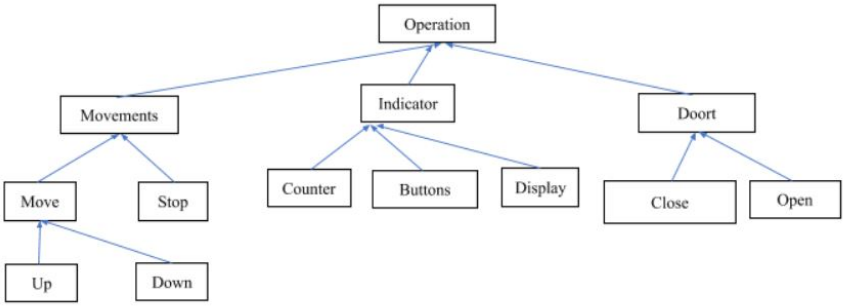

Fig. 1. Modularization of Elevator System, Module : Operations.

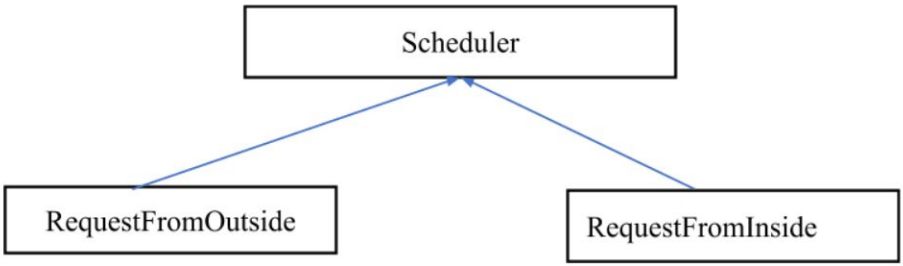

Fig. 2. Modularization of Elevator System, Module : Scheduler.

- Operations - This module is the base of all operations that can be performed by client, which will be denoted as services for the client. Operation is further modularized into Movement, Indicator and Door. This module hides all working details like opening and closing the door, indicator mechanics and so on, which help an elevator unit to interact with environment.
- ElevatorUnit System - this module deals with the basic hardware component of an elevator: Lights, Door, and Motor. This module hides the details about how to make these services user serviceable.

Figure 1,2,3 shows a Modularization of an Elevator System, when combined.

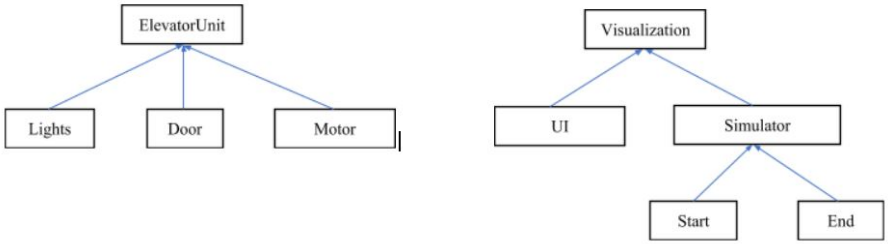

Fig. 3. Modularization of Elevator System, Module : Visualization and Elevator Unit.

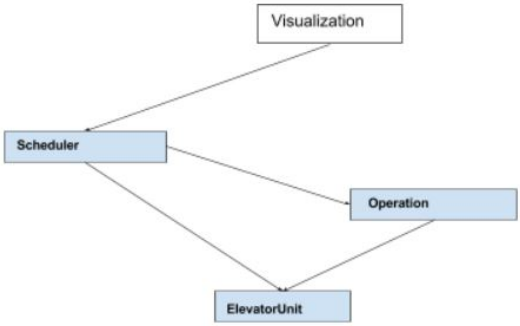

Fig. 4. USES relation among Components, this doesn't include elementary level relation

USES Relation among Modules : Now, for one important part of any software architecture, the dependency relation among modules, in this subsection we will try to give general ideas about how modules are being used in order to provide software service to client.

Figure 4 shows a Uses relation among Components.

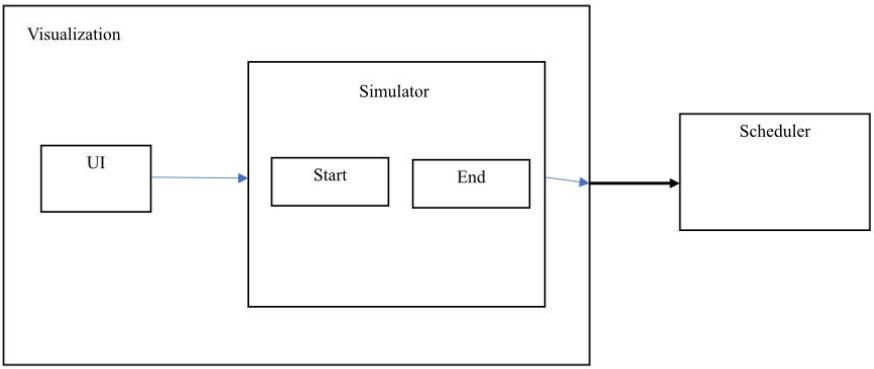

Fig. 5. GDN for Component Visualization,

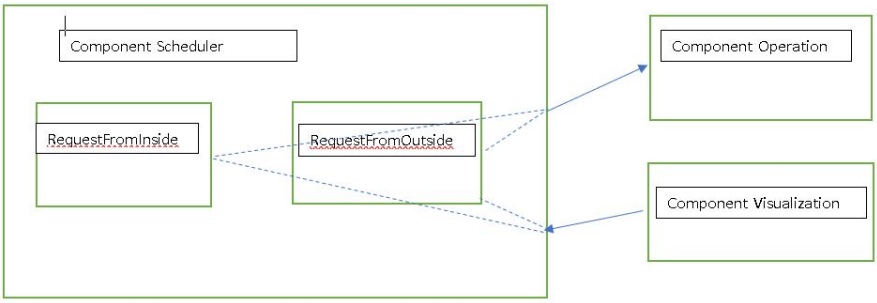

Fig. 6. GDN for Component Scheduler,

2. Design Details

For an Elevator system, after describing Components, it is important to describe Design. We'll briefly give idea of Design using GDN for every components and it's interaction in Uses Relation described in previous section

- Visualization : Refer Fig. 5
- Scheduler : Refer Fig. 6
- Operation : Refer Fig. 7
- ElevatorUnit : Refer Fig.8

After describing Components design details using GDN diagrams, We would like

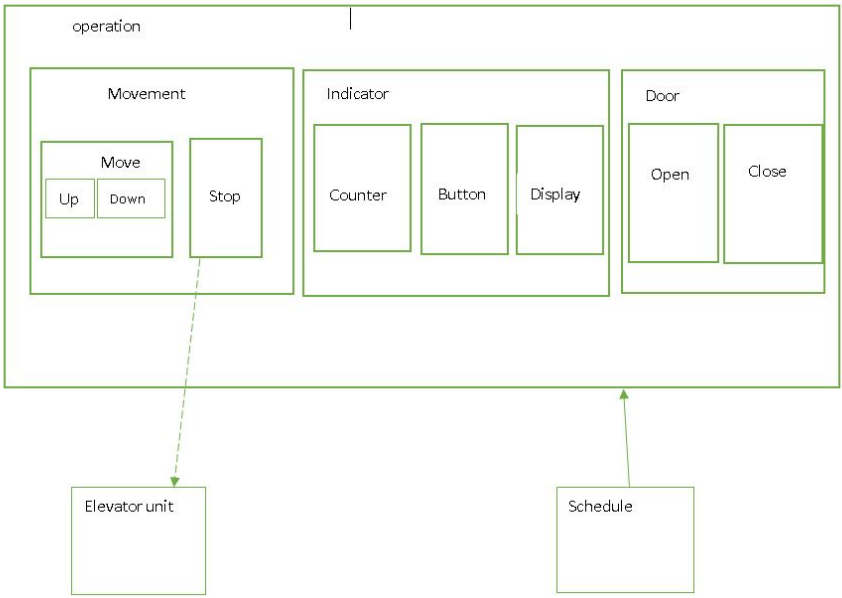

Fig. 7. GDN for Component Operation,

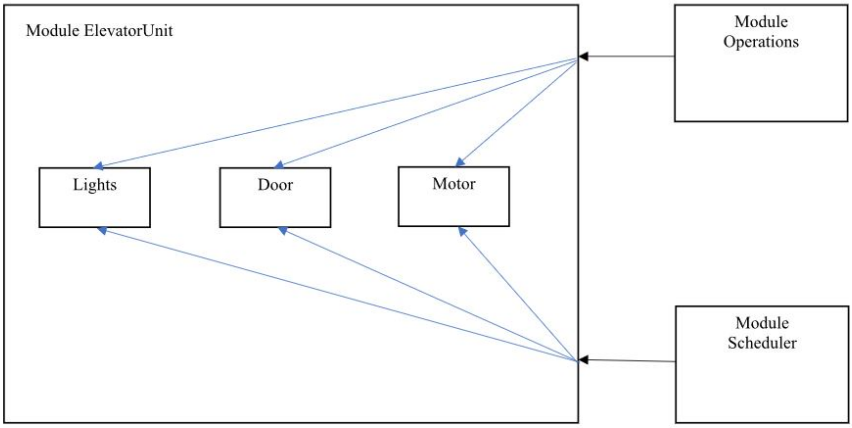

Fig. 8. GDN for Component ElevatorUnit,

to mention few interfaces(in Java Language) per module which includes only method signature.

- Visualization
  - Simulator
    - void initiate()
    - int end()
    - int createLift()
    - int performOperation(int buttonPressed)
  - UI
    - void ButtonPress()
    - int start()
    - boolean createLifts(int numberOfLifts)
    - int end()
- Scheduler
  - boolean scheduleRequest(int floor, boolean direction)
  - boolean scedhuleRequestfromInside(int liftID, boolean direction)
- Operation
  - Movement
    - boolean moveUp()
    - boolean moveDown()
    - boolean stop()
  - Indicator
    - int checkCounter(int liftID)
    - int displayScreen(int liftID)
    - int buttonPress(int typeOfButton)
  - Door
    - boolean openDoor(int liftID)
    - boolean openDoor(int liftID)
- ElevatorUnit
  - Lights
    - boolean isLightOn(int buttonId)
    - boolean turnOff(int buttonId)

    * boolean turnOn(int buttonId)
  - Motor
    * void beginMotor(int liftID)
    * void shutMotor(int liftID)
  - Door
    * boolean isDoorOpen(int liftID)

3. Troubleshooting

For any problem please contact the editor

karma,mingling,zyliu@udel.edu

Acknowledgements

This is a Lab report created as an artifact for design of an elevator system, under guidance Professor Stephen Seigel.