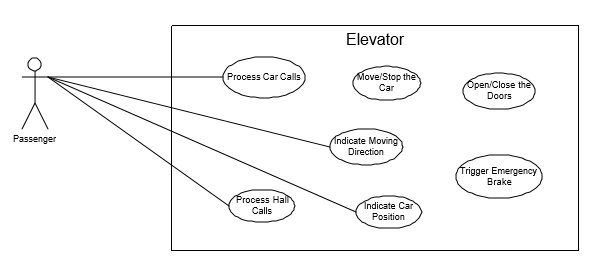
**# SystemDesingProgram**

**Use case diagram**

All systems interact with human or automated actors that use the system for some purpose, and both human and actors expect the system to behave in predictable ways.



**Figure 1: Use Case Diagram of Elevator System**

There are seven use cases based on the current requirement of the elevator system, as shown in Figure 1:

**Process Car/Lift Calls**: This use case includes several scenarios, which will be described in detail in later sections of this paper. These scenarios includes that the elevator receives lift/car calls from the passengers, turns on or turns off the light of lift/car call buttons, updates the record of lift/car calls stored in system controlling parts, etc.

**Process Hall Calls**: Similar to Lift/Car Call processing, this use case includes that the elevator receives hall calls from the passengers, turns on or turns off the light of hall call buttons, updates the record of hall calls in system controlling parts, etc.

**Move/Stop the Car**: The main function of an elevator will include the changing of driving speed, how to make the decision of stop, and driving directions of the car.

**Indicating Moving Direction:** The elevator should have this mechanism to let the passengers know the current moving direction of the car such that the passenger might decide whether to enter the car or not.

**Indicating Car Position:** Similarly, the elevator should let the passengers know whether his/her destination floor is reached so that the passenger may decide to leave the car.

**Open/Close the Doors:** The elevator should be able to open and close the doors for the passengers to get in and out of the car. The functional areas of this use case should also enable the passengers to make door reversals when the doors are closing and the passenger wants to get in the car.

**Trigger Emergency Brake:** There is safety mechanism within the car to make sure that an unsafe state is not transiently generated.

With this much of understanding of elevator system, we can start identifying the classes required it to design a system.

**Identifying Classes/Objects:**

**Control objects**

**ElevatorControl:** The central controlling object in the elevator system. ElevatorControl communicates and controls all other objects in the system.

**Door:** There are two doors in the system, the “god” object — the ElevatorControl — command the doors to open and close.

**Car/Lift:** The car is being controlled to move up and down (in different speeds), to make stops at floors when necessary.

**Button:** The ElevatorControl class also controls the button class, which further generalizes two subclasses CarCallButton and HallCallButton. The control object communicates with the Button objects, get the information whether a button is pressed and in turn controls the illumination of Button lights.

**Indicator:** There are two kinds of indicators in the system, the CarPositionIndicator and the CarDirectionIndicator (i.e. the CarLantern). The indicators are controlled to show the information about the current position and moving direction of the car.

**Safety:** Whenever an emergency happens, the ElevatorControl commands the Safety

**Controller classes:**

DoorControl controls the action of DoorMotor. DoorMotor can be commanded to open, close, or make a door reversal.

DriveControl controls the elevator Drive, which acts as the main motor moving the car up and down, and stopping at floors when necessary.

LanternControl are in the number of two, CarPositionIndicator and the CarDirectionIndicator.

HallButtonControl exists in pair on each floor, where one controls the Up HallCallButton and the other Down. The HallButtonControl accepts hall call button presses controls as well as gives feedback to hall call lights.

CarButtonControl is one for each floor and all locate in the lift/car. The CarButtonControl accepts CarCallButton calls and is in charge of turning on/off the corresponding lift/car call lights.

CarPositionIndicator gives value to the CarPositionIndicator so that the passengers might know the current position of the lift/car.

Dispatcher does not control actual elevator components, but it is important in the software system. There is one Dispatcher for each lift/car, whose main function is to calculate the target moving direction and destination for the lift/car, as well as to maintain the opening time for the doors. The Dispatcher interacts with nearly all the control objects in the system except for LanternControl.

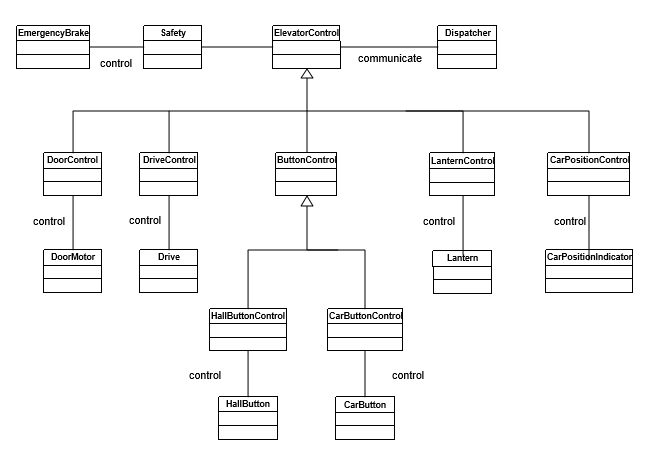


Figure 2: Class Diagram — the Software Architecture View

Hope everything is going well till now, let’s understand everything with some of sequence diagrams.

**Sequence flow**

It may be hard to explain verbally but if you clearly understand how its going to work, you can draw some sequence diagrams and try to explain the flow with some examples in parallel.

**Process hall calls :** There are two scenarios for this use case:

1.1 the elevator is moving towards the same direction as the passenger’s destination.

1.2 the elevator is moving towards the opposite direction as the passenger’s destination.

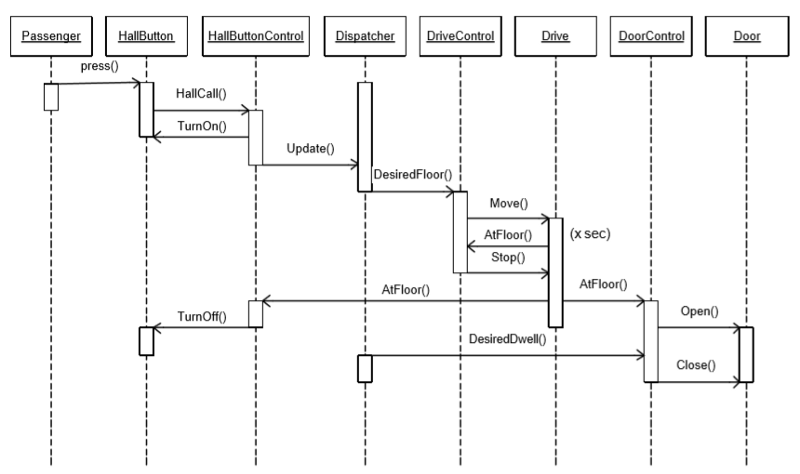


Figure 3: 1.1&1.2 — Hall Call Service

**2. Move/Stop the car:**

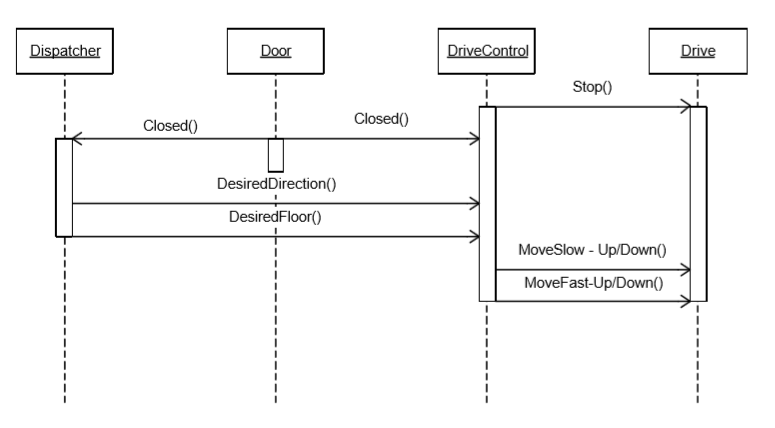


Figure 4: Moving the Car from Fast to Slow then to Stop

**3. Indicate moving direction:**

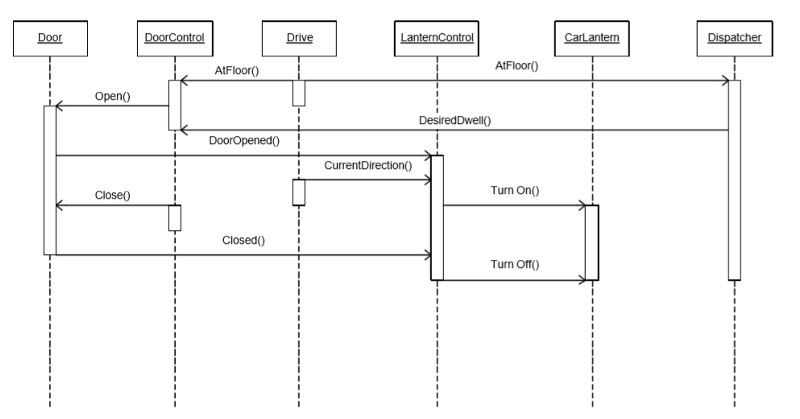


Figure 5: Indicating the Moving Direction

**4. Trigger emergency brake :** There are five scenarios for this use case:

1. If the car is commanded to stop but it won’t stop at a desired floor, the emergency brake will be triggered.

2. If the car is commanded to move but it does not move, the emergency brake will be triggered.

3. If the doors are commanded to open when the car stops at a floor, but the doors won’t open, the emergency brake will be triggered.

4. If the doors open when the car is moving, the emergency brake will be triggered.

5. If the car keeps going when the hoist way limit is reached, the emergency brake will be triggered.

I will be showing only one sequence diagram for case 5.

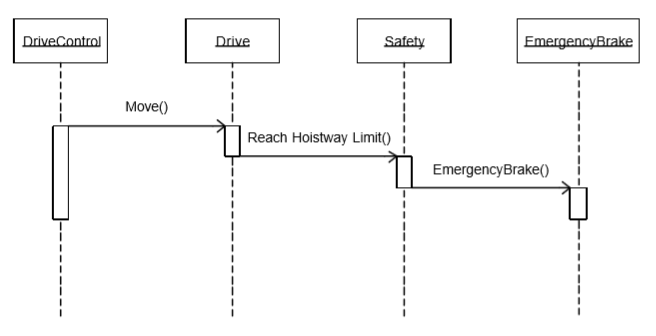


Figure 6: Emergency Brake — The elevator keeps going when hoistway limit is reached

I am expecting most of the things will become picture in your mind and easy to remember as well as explain. For backend design , please follow System Design series