Elevator System

Problem definition

An **elevator** is an integral part of buildings that have multiple floors. The elevator car could be in different states, either up or down, or could be stopped on some floor. Anyone can request an elevator car from any floor using the buttons on the panel. The elevator car's algorithm will set the priority and take action accordingly, so the wait time is minimum. Inside an elevator, there will be a panel for passengers to select the floor on which they want to go. The elevator car will have a fixed capacity for the number of passengers and a display to show on which floor the elevator car is currently located.

Expectations from the interviewee

Numerous components are present in a typical elevator system, each with specific constraints and requirements placed on them. The following provides an overview of some of the main expectations that the interviewer will want to hear you discuss in more detail during the interview:

Multiple elevators

You can also ask the interviewer whether the system should handle multiple elevators or just the single one. For this you can ask the following questions:

1. Can there be multiple elevator cars in the building?
2. How could a one-elevator system be different from a multi-elevator system in terms of user wait time and running cost?

Display

You may want to ask the interviewer about the display of the elevator system:

1. How can passengers see the status of the elevator car and request an elevator car?
2. Would the display be the same inside and outside of the elevator?

Optimization

An interviewer would expect you to ask questions about the optimization of the elevator system in terms of wait time, maintenance, throughput, etc. You can ask questions like:

1. What would be an optimized solution to minimize the wait time of the passengers?
2. How are we going to minimize the running cost of the elevator system?

Design approach

We'll design this elevator system using the bottom-up design approach. For this purpose, we will follow the steps below:

- Identify and design the smallest components first, like, the button and door.
- Use these small components to design bigger components, for example, the panel, elevator car, and building.
- Repeat the steps above until we design the whole system.

Requirements for the Elevator System

For the elevator design problem, the requirements are defined below:

R1: There exist multiple elevator cars and floors in the building.

R2: The building can have a maximum of 15 floors and three elevators.

R3: The elevator car can move up or down or be in an idle state.

R4: The elevator door can only be opened when it is in an idle state.

R5: Every elevator car passes through each floor.

R6: The panel outside the elevator should have buttons to call an elevator car and to specify whether the passenger wants to go up or down.

R7: The panel inside the elevator should have buttons to go to every floor. There should be buttons to open or close the lift doors.

R8: There should be a display inside and outside the elevator car to show the current floor number and direction of the elevator car.

R9: The display inside the elevator should also show the capacity of the elevator car.

R10: Each floor has a separate panel and a display for each elevator car.

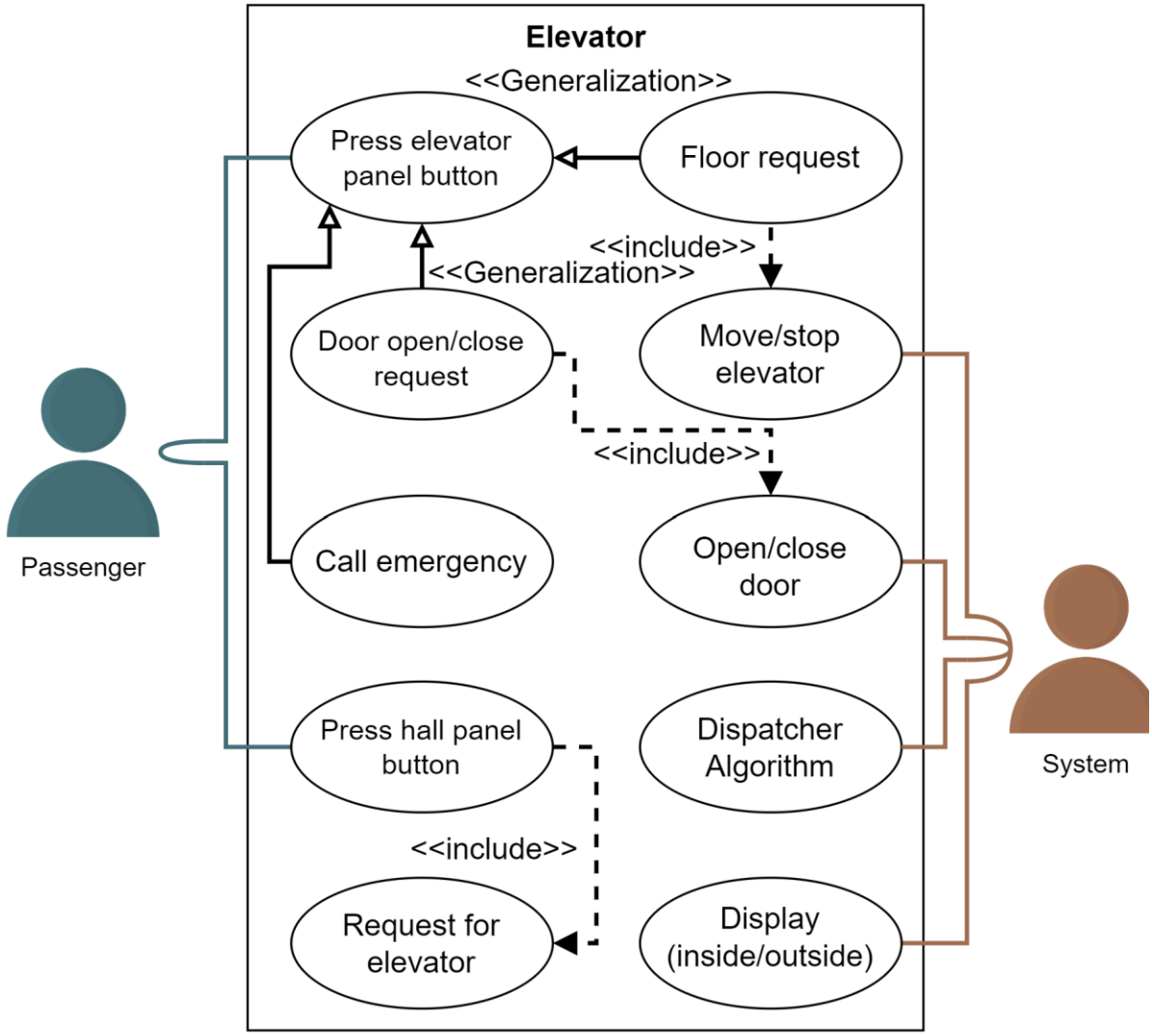
R11: Multiple passengers can go to the same or different floors in the same or opposite direction.

R12: The elevator system should be able to control the elevator car movement and the door functioning and monitor the elevator car.

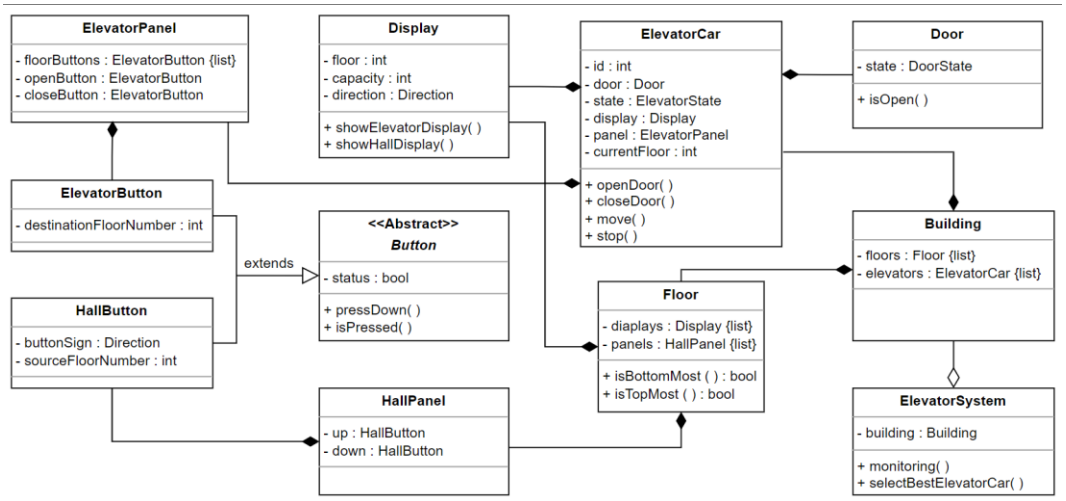
R13: The elevator control system should be able to send the most appropriate elevator to the passenger when the passenger calls the elevator car.

R14: The elevator car can carry a maximum of eight persons or 680 kilograms at once.

Use case diagram



Class Diagram for the Elevator System



Additional requirements

The interviewer can introduce some additional requirements in the elevator control system, or they can ask some follow-up questions. The additional requirement can be about how the dispatcher works in the elevator system. The interviewer can ask to devise an algorithm to optimize any of these:

- To minimize the wait time of the system
- To minimize the wait time of the passenger
- To maximize throughput
- To minimize the power usage or cost

To optimize the elevator system, we have different dispatching algorithms.

FCFS

First Come First Serve (FCFS) is a scheduling algorithm by which the passenger who comes first gets the elevator car and reaches the destination. There are four states of an elevator car with respect to the passenger:

- The elevator car is in an idle state.
- The elevator car is moving towards the passenger and in the same direction the passenger wants to go.
- An elevator car is moving towards the passenger but in the opposite direction the passenger wants to go.
- The elevator car is moving away from the passenger.

In this algorithm, the dispatcher will try to find elevators that are in either of the first two states and ignore those elevators which are in either of the last two states.

The advantage of this algorithm is that it is simple and easy to implement. The drawback of this algorithm is that extra elevator movements occur by this algorithm which results in more power usage and cost. To implement FCFS, we can use a queue data structure to keep track of which passenger comes first.

SSTF

Shortest Seek Time First (SSTF) is an algorithm in which the passenger who is closest to the elevator car would get the elevator car. This algorithm is considered better than FCFS since less elevator movement is required as compared to the FCFS algorithm. This algorithm also results in an increased throughput. However, there is a loophole in this method where it always chooses the minimum distant passengers and ignore the farther ones completely. To implement this algorithm, we can use a priority queue, min-heap, or an array data structure.

SCAN

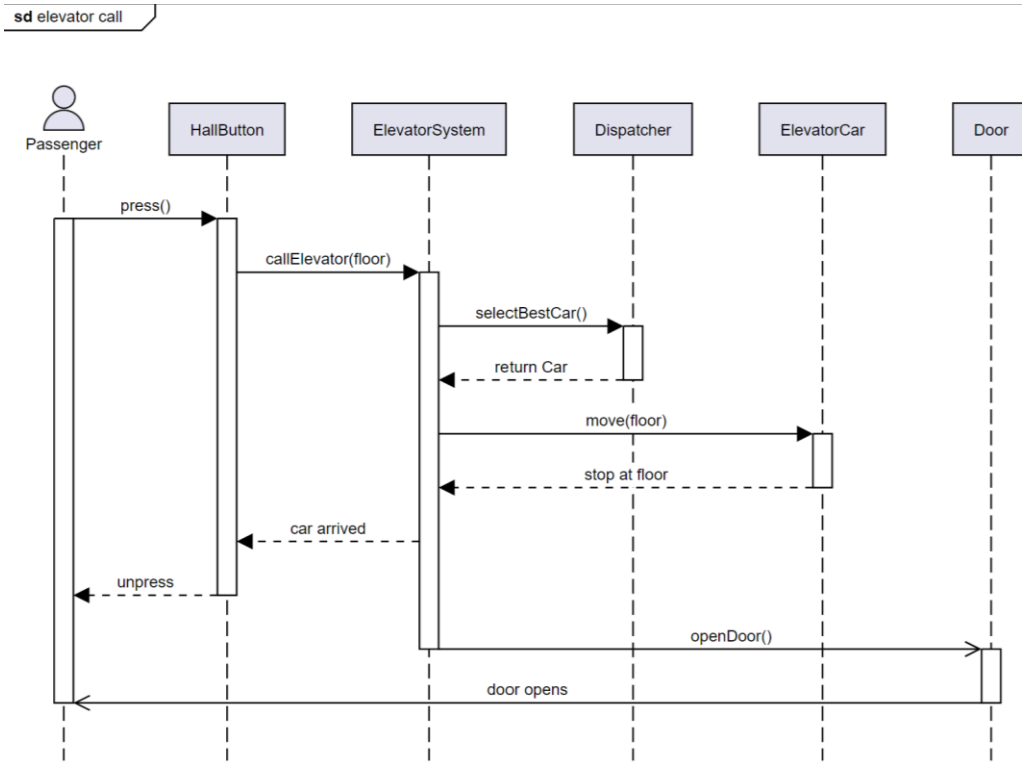
SCAN is also known as the **Elevator Algorithm**. The elevator car starts from one end of the building and moves towards the other end, servicing requests in between. The advantage of this method is that it serves multiple requests in parallel. However, it results in increased cost as the elevator car only changes its direction at either the top floor, or the lowest floor. The implementation of SCAN can be done using two boolean arrays or a single HashMap, or two priority queues data structures to track the floor where the elevator should stop.

LOOK

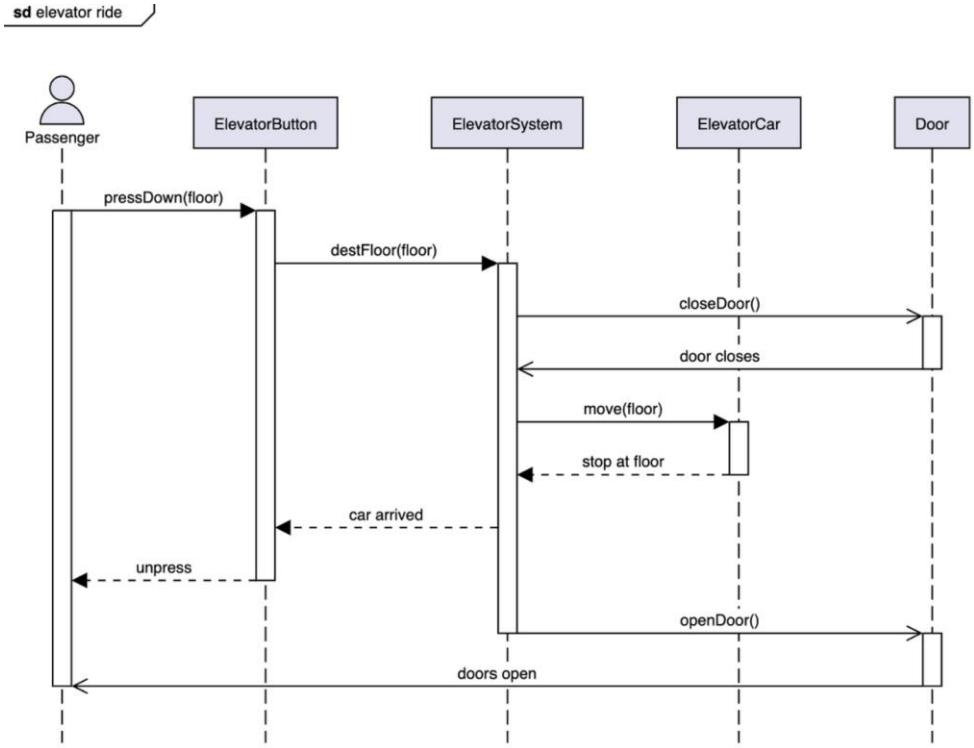
LOOK is also known as the look-ahead SCAN algorithm. It is an improved version of the SCAN Algorithm. In this algorithm, the elevator car stops when there is no request in front of them. It will move again on the basis of the request. The advantage of this algorithm is that the elevator car does not always go till the end of the building but can change its direction in between. This algorithm can be implemented using a HashMap, TreeMap, or binary search tree data structure.

Sequence Diagram for the Elevator System

Elevator call

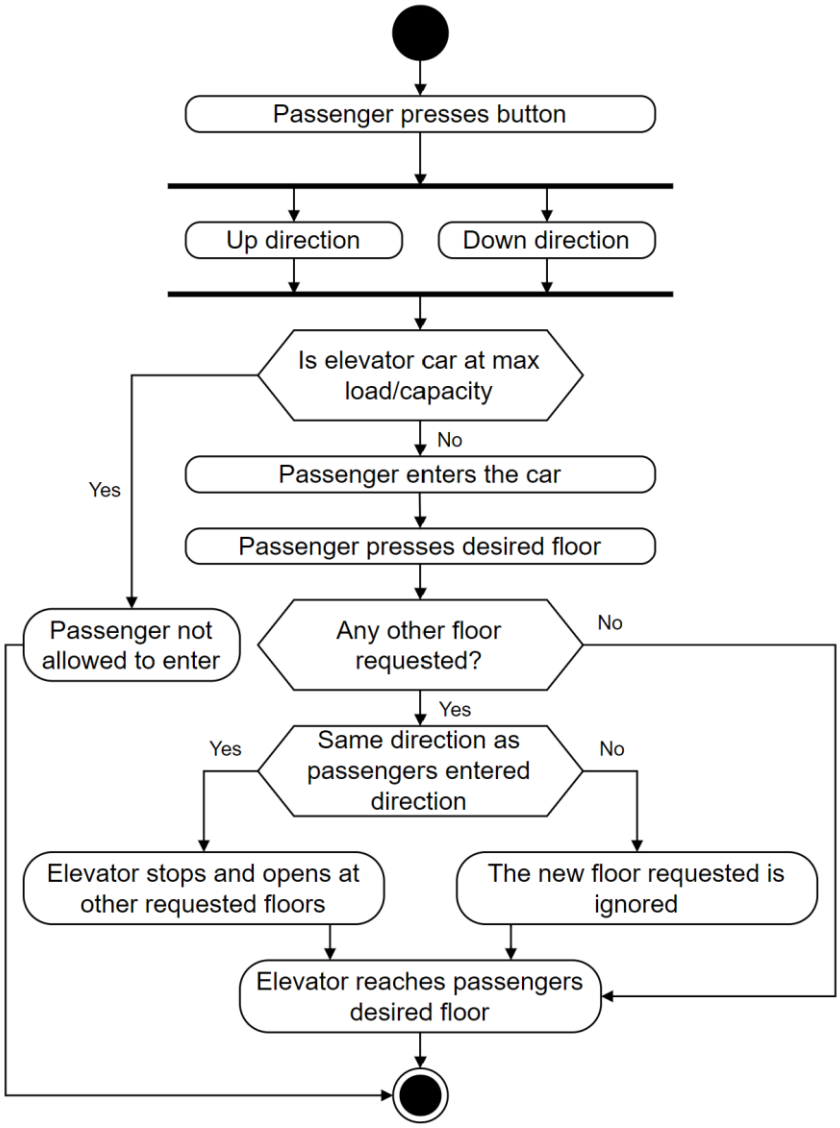


Elevator ride

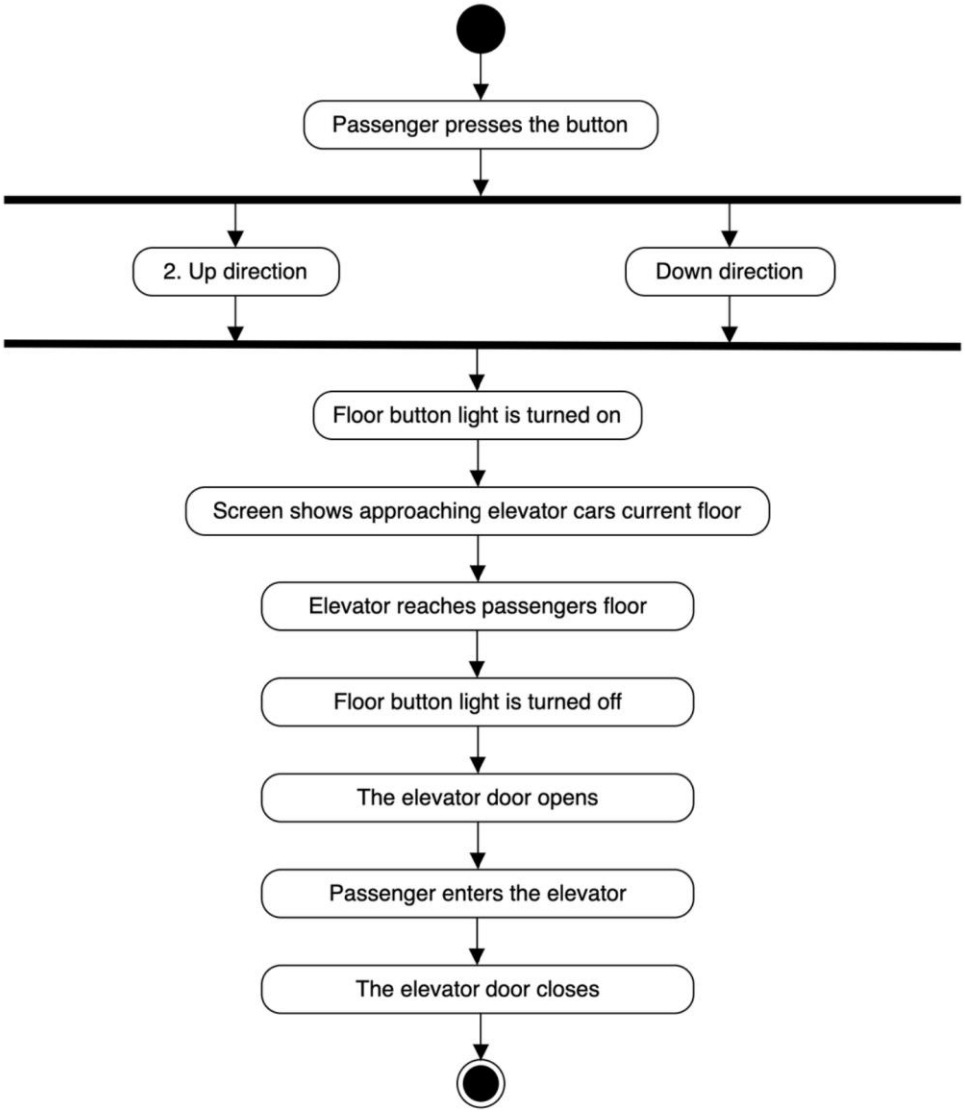


Activity Diagram for the Elevator System

The passenger arrives at the desired floor



The passenger calls for the elevator


Passenger presses the button
2. Up direction
Down direction
Floor button light is turned on
Screen shows approaching elevator cars current floor
Elevator reaches passengers floor
Floor button light is turned off
The elevator door opens
Passenger enters the elevator
The elevator door closes