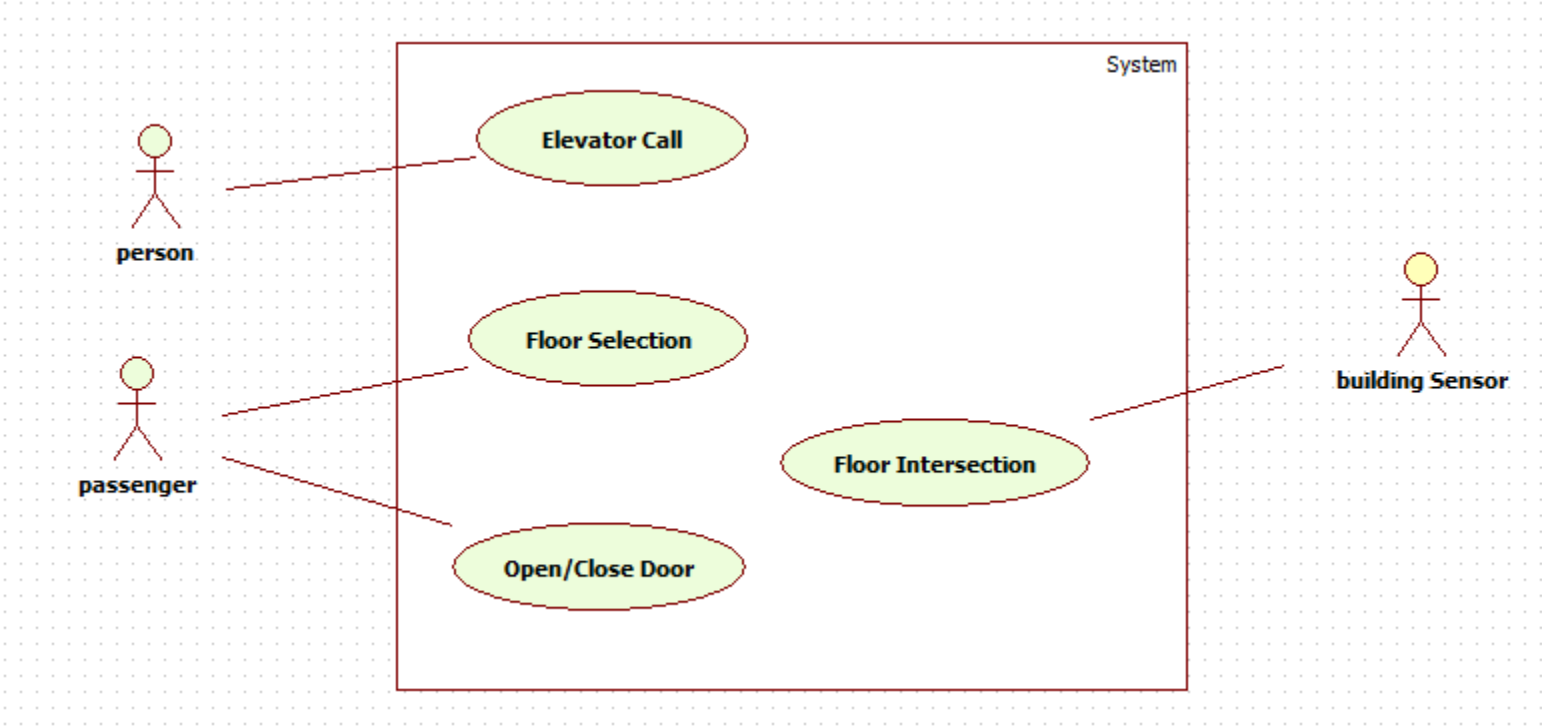
ELEVATOR CONTROLLING SYSTEM

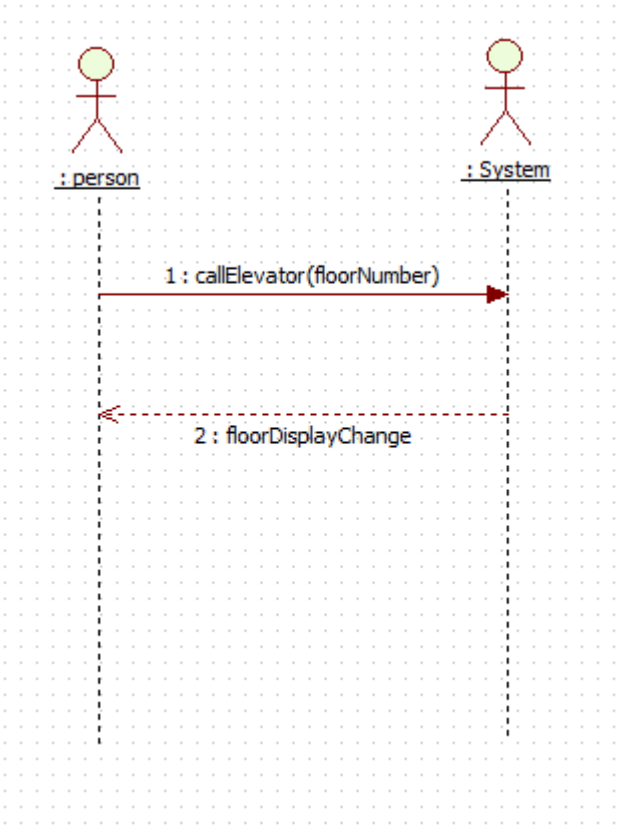
Content

i. Use Case Model: Use Case Diagram



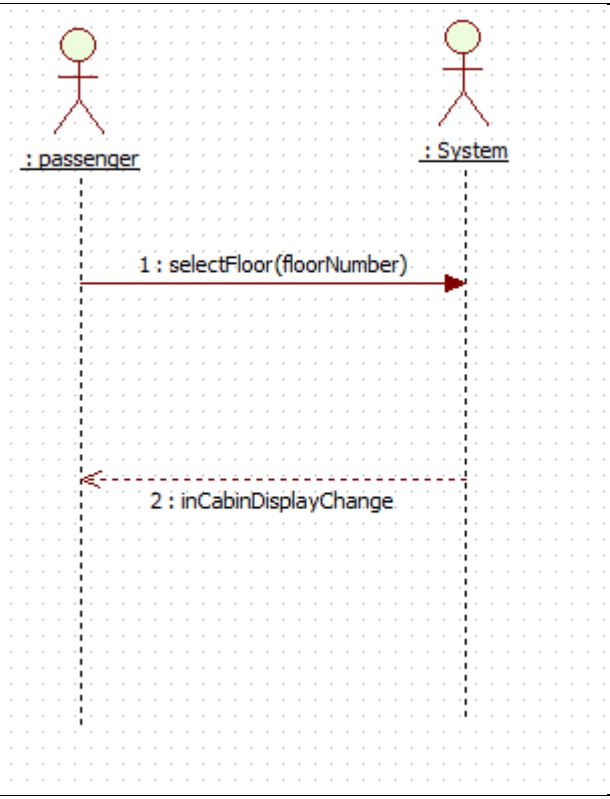
ii. Use Case Model : Text Use Cases and System Sequence Diagram

UC1: Elevator Call

1. Person approaches an elevator and requests a direction by pressing corresponding floor button. 2. The button sends a signal to Elevator Controller. The emitted event passed to scheduler, which arranges new request according to its type adding to the Upwards or Downward lists. 3. Scheduler from the list according to the Elevator's current direction selects the nearest job by destination and if it is nearer than Elevator's current job destination, change the job, and add previous one to the appropriate list (updates the job). Otherwise do nothing. If elevator was idle, pick the nearest job list. 4. If Elevator is idle put in movement elevator. Otherwise continue path. 5. Elevator in motion is displaying his current destination and direction. The currFloorNumber changes continually during motion. And Elevator status sets to waiting. 6. Elevator reaches the requested floor. 7. Engine Stops at the destination. 8. Floor Display shows Elevator's current destination and direction. 9. Elevator and floor doors are opened. Person becomes passenger entering to the cabin.	

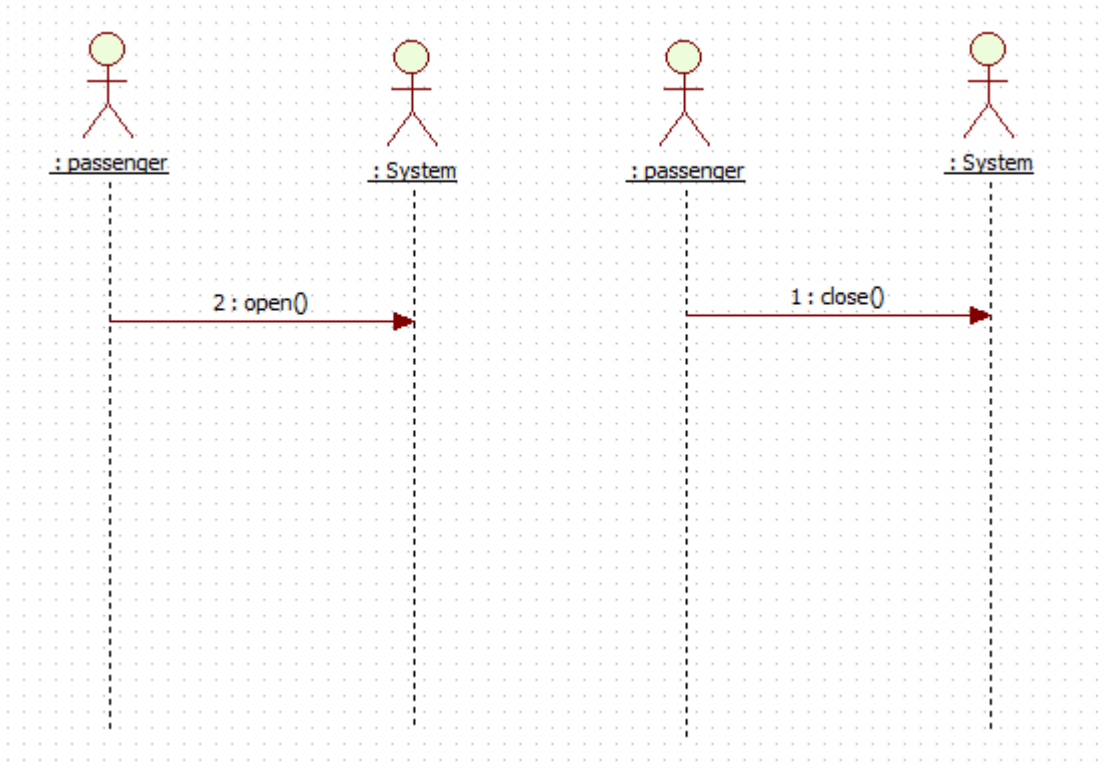
UC2: Floor Selection

1. Passenger selects the floor number.
2. Controller notified about floor selection, decide the job direction based on Elevator's current direction and pass job to Scheduler.
3. Job added to corresponding List. Scheduler updates Elevator's job if necessary,
4. Elevator will perform selected job, meaning that it will arrive to the corresponding floor.
5. Floor and elevator doors are opened.
6. In cabin Display shows continually the elevator's destination and direction of the motion.
7. Scheduler updates Elevator job.



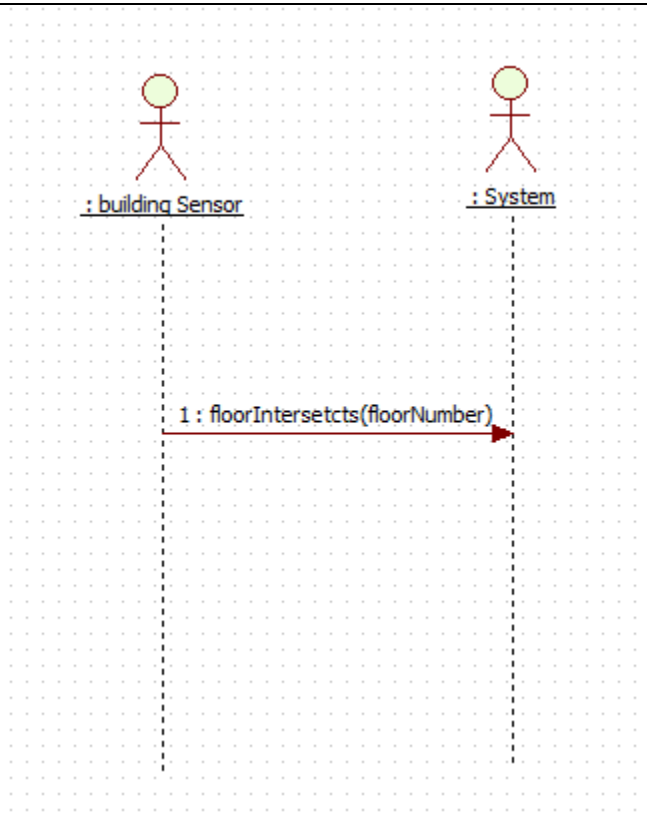
UC3: Open/Close Door

1. The passenger selects open/close door button.
2. If the elevator status is waiting (elevation is in waiting mode when it stops and the timeOut isn't finished), will force to open/close the floor and elevator doors.



UC3: Floor Intersection

1. Elevator is arriving to the floor.
2. The building sensor system is activated. This in its turn activates Elevator sensor.
3. Elevator sensor sends floorIntersection signal to Elevator identifying the floor number.
4. If the intersected floor is destination, Engine stopped.
5. Elevator status is waiting until timeOut is over or user invoked close door event.
6. Floor and elevator Doors are opened.
7. Display information is updated.



iii. Use Case Model: System Operation Contracts

Contract C1:

Operation: callElevator(floorNumber, direction)
References: Elevator Call use case

Preconditions:

- fb : floorButton linked with elS: ElevatorSystem. fb.isActive = true and fb.floorNumber = floorNumber.
- There is an el : Elevator in a working condition linked with sch: Scheduler and elS: ElevatorSystem, moving by el.direction = dir direction.
- el : Elevator linked with eng: Engine, and having el.status = status condition.

Post conditions:

- elS determine request direction by elevator currentPos and requested floorNumber, assigns request to sch: Scheduler.
- If in the sch:Scheduler appropriate list there is job with between el.currentFloor and el.destination change jobs: store el.destination in sch and update el.destiantion = newDst. If el.destination is empty just el.destination = newDst.
- Move elevator to the slected floor. If el.status is Idle turn on Engine resetting clock. Engine moves elevator to the requested floor and stops it at the destination point.
- Elevator and floor doors are opened. el.status becomes “waiting” in timeOut periodInCabinDisplay changes during the motion to the currentFloorNumber and motion direction. el.currentFloorNumber = el.destination is job’s floor Number.

Contract C2:

Select Floor and call floor use cases post conditions are the same, as each time we just invoke elevator movement.

Operation: selectFloor(floorNumber)
References: Floor selection use case.

Preconditions:

- There is an instance of inButton : Button linked with elev: Elevator and elevSys: ElevatorSystem, for which inButton.isActive is true
- A instance of elev : Elevator with currentFloor, status and destinationFloor values which is linked with Scheduler
- And the elevSys:ElevatorSystem which also linked with sch: Scheduler with instances up:UpWardLists and down:DownWardLists

Post conditions:

- Scheduler determine the direction of requested floor relative to current direction. Add the floorNumber to appropriate destination lists: up:UpwardLists or down:DownWardLists.
- The top destination from appropriate list is selected. If is nearer than elev.destination then change destination values:
  Remove top destination from list, put elev.destination.
- elev.destination = removed destination.
- Engine moves the Elevator to the destination floor number assigned by job.

Contract C3:

Operation: openDoors()
References: Open/Close Doors use case

Preconditions:

- There is an instance of elev:Elevator linked with elevSys: ElevatorSystem , for which elev.status = waiting.
- ed:Door instance linked with elev:Elevator, for which ed.status = notOpen.
- fd:Door instance linked with fl:Floor, for which fd.status = notOpen.
- fl.number = elev.currentFloor.
- each Door instance also linked with their engines.

Post conditions:

- fd:Door is opened, by fDE:DoorEngine, fd.status = open,
- ed:Door is opened, by eDE:DoorEngine ed.status = open.
- fDE.remainingTime = fd.timeOut and eDE.remaningTime = eDE.timeOut

Contract C4:

Operation: closeDoors()
References: Open/Close Doors use case

Preconditions:

- There is an instance of elev:Elevator linked with elevSys: ElevatorSystem , for which elev.status = waiting.
- ed:Door instance linked with elev:Elevator, for which ed.status = notOpen.
- fd:Door instance linked with fl:Floor, for which fd.status = notOpen.
- fl.number = elev.currentFloor.
- each Door instance also linked with their engines.

Post conditions:

- fd:Door is closed, by fDE:DoorEngine, fd.status = open,
- ed:Door is closed, by eDE:DoorEngine ed.status = open.
- fDE.remainingTime = 0 and eDE.remaningTime = 0

Contract C5:

Operation: floorIntersection(floorNumber)
References: Floor Intersection use case

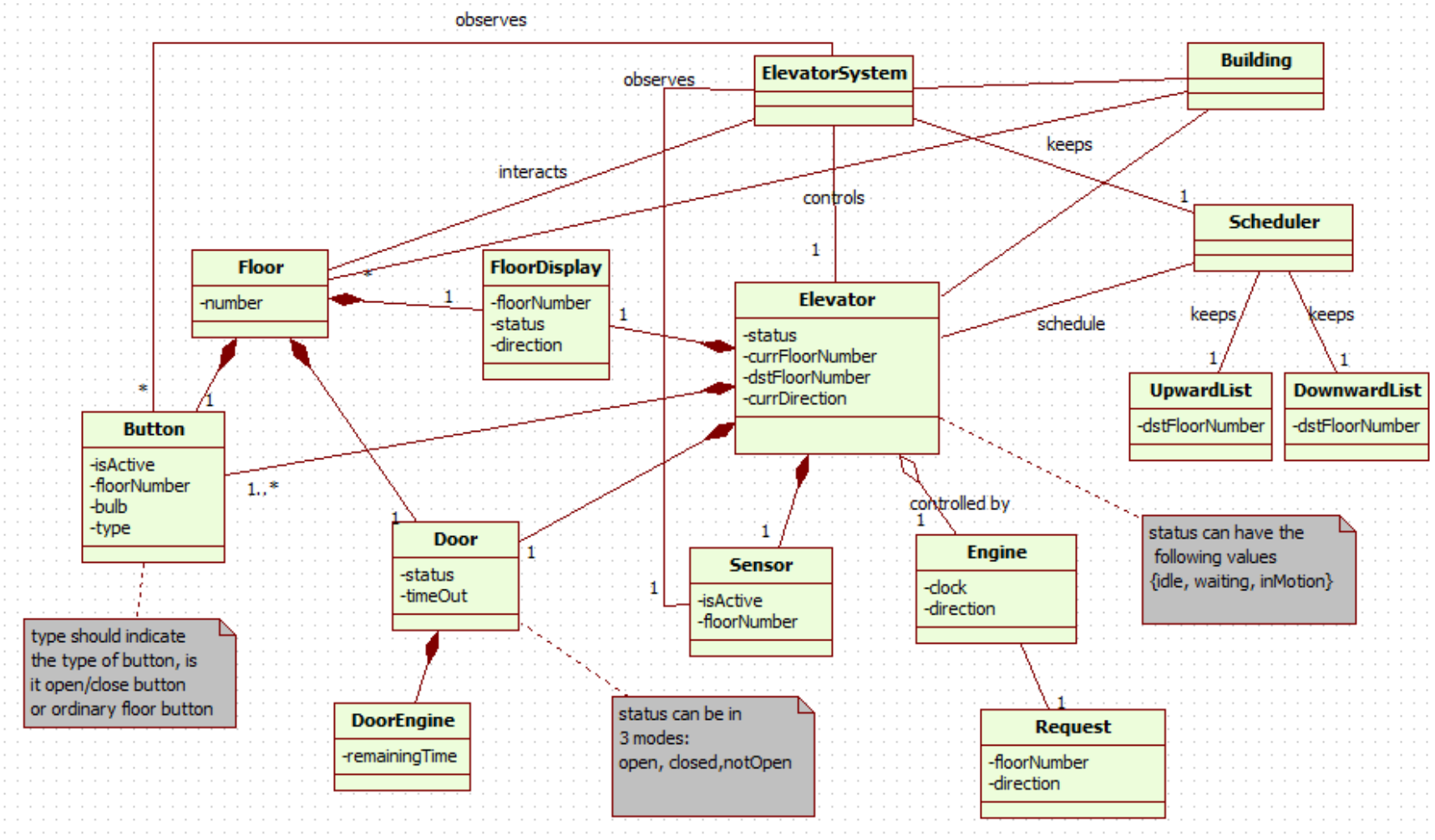
Preconditions:

- There is an instance of elevSensor: linked with elev: Elevator.
- ed:Door instance linked with elev:Elevator, for which ed.status = closed.
- fd:Door instance linked with fl:Floor, for which fl.status = closed.
- each Door instance also linked with their engines : fDE:DoorEngine and eDE:DoorEngine respectively.
- fDisp:Display instance linked with fl:Floor
- eDisp: Display instance linked with el:Elevator

Post conditions:

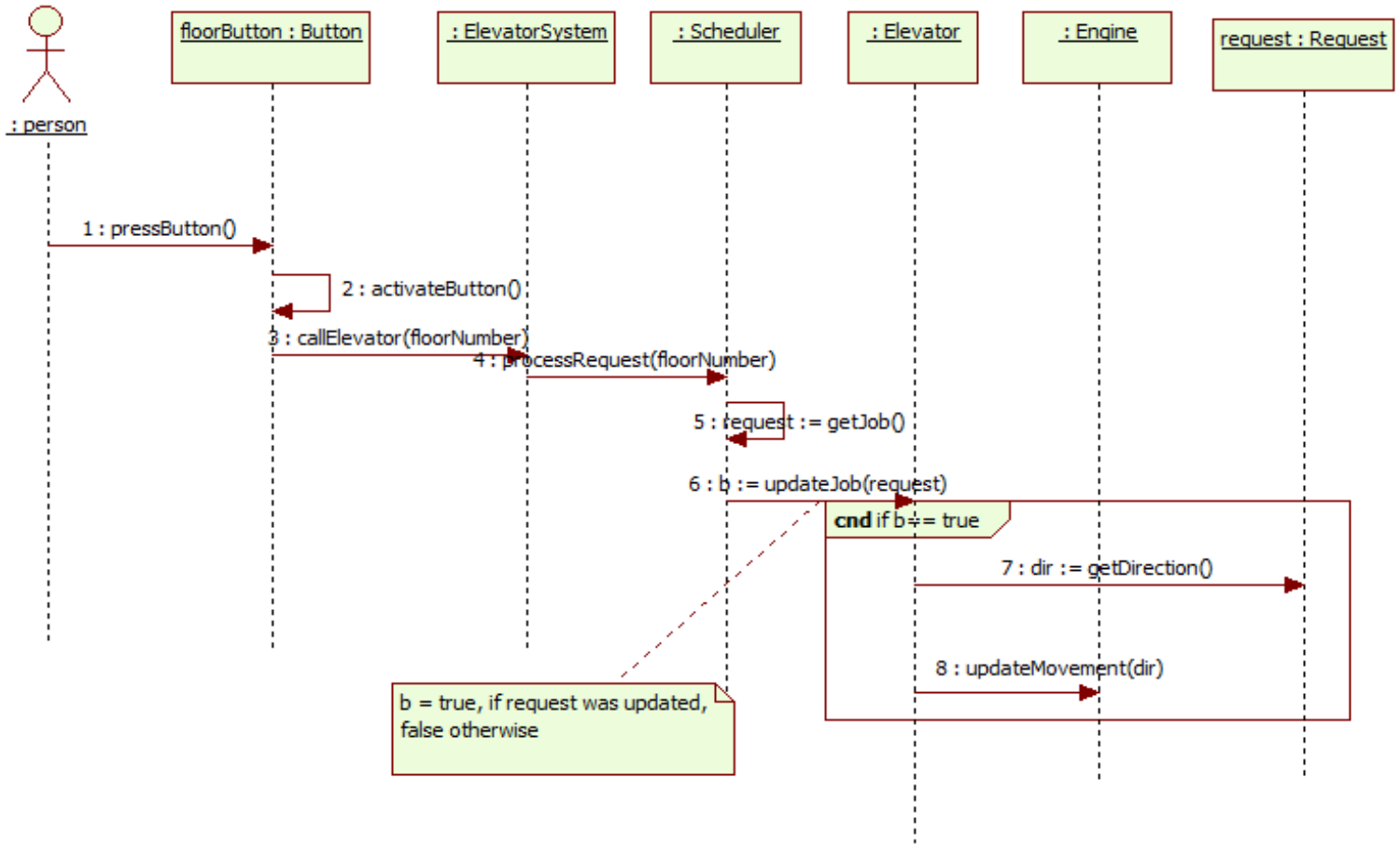
- fDisp.status and eDsip.status are setted as active
- fDisp. floorNumber = floorNumber, eDisp .floorNumber= floorNumber
- fDisp.direction = eDisp.direction = fl.direction
- if el.dstFloorNumber == floorNumber fd.Door is opened by fDE:DoorEngine, fd.status = open, fd:Door is closed, by fDE:DoorEngine fd.status = open

2. Domain Model

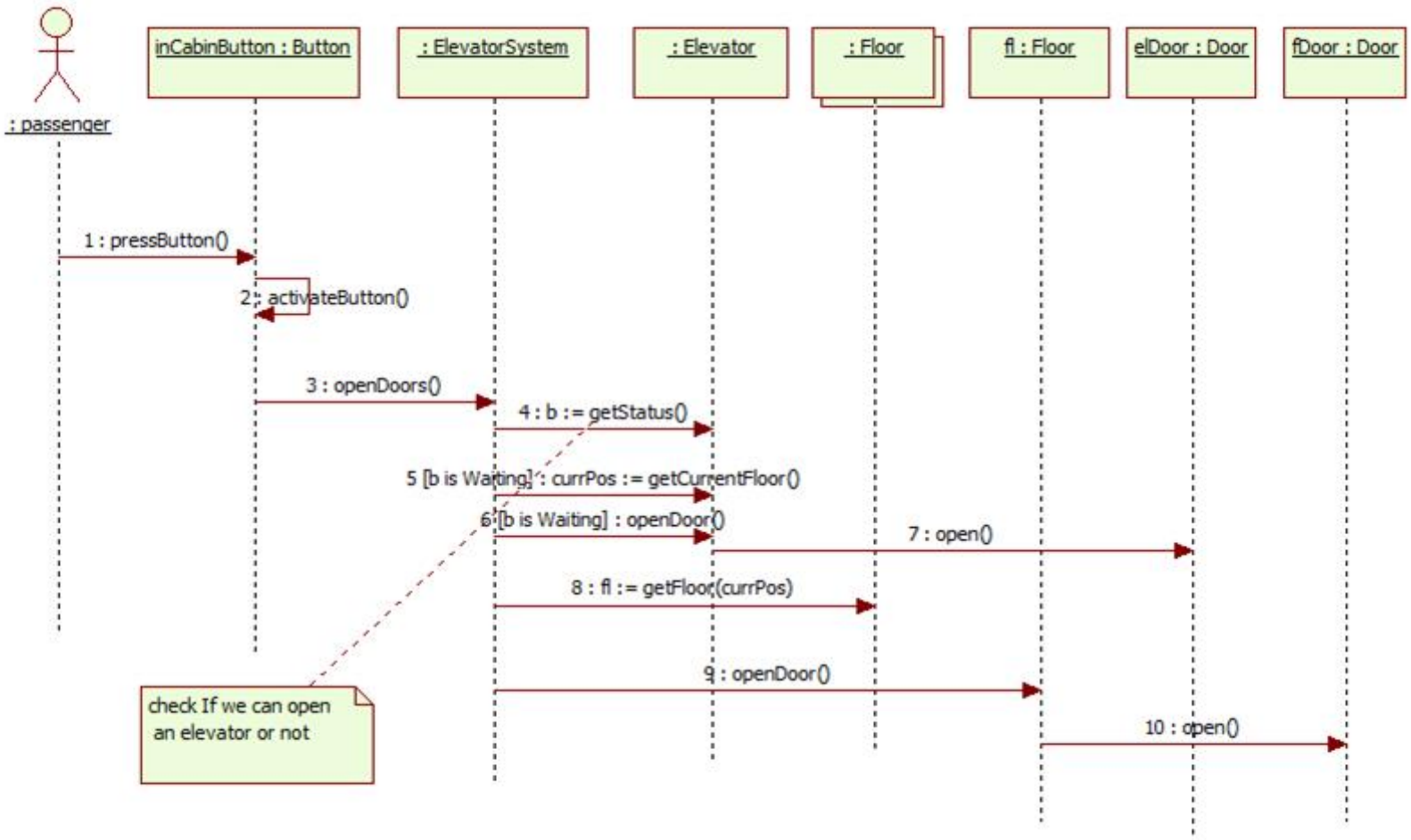


3.Design Model - Interaction Diagrams

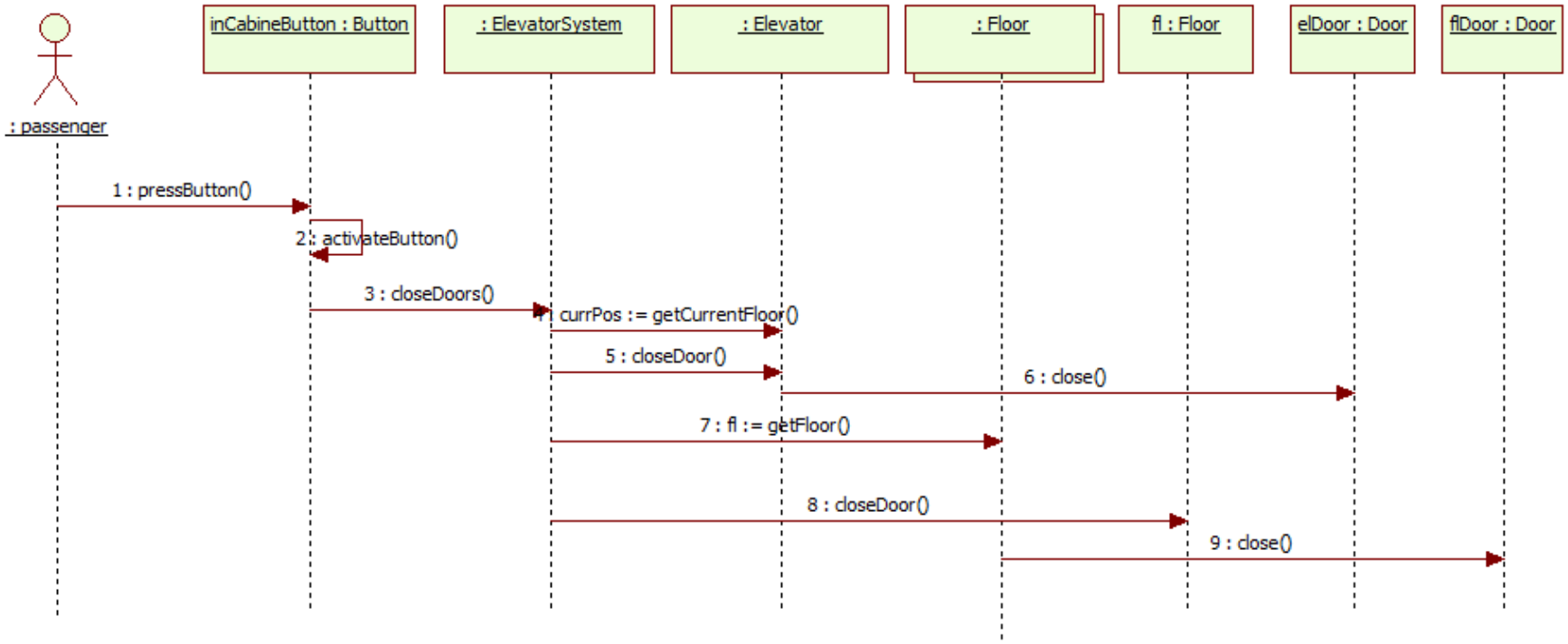
Call Event



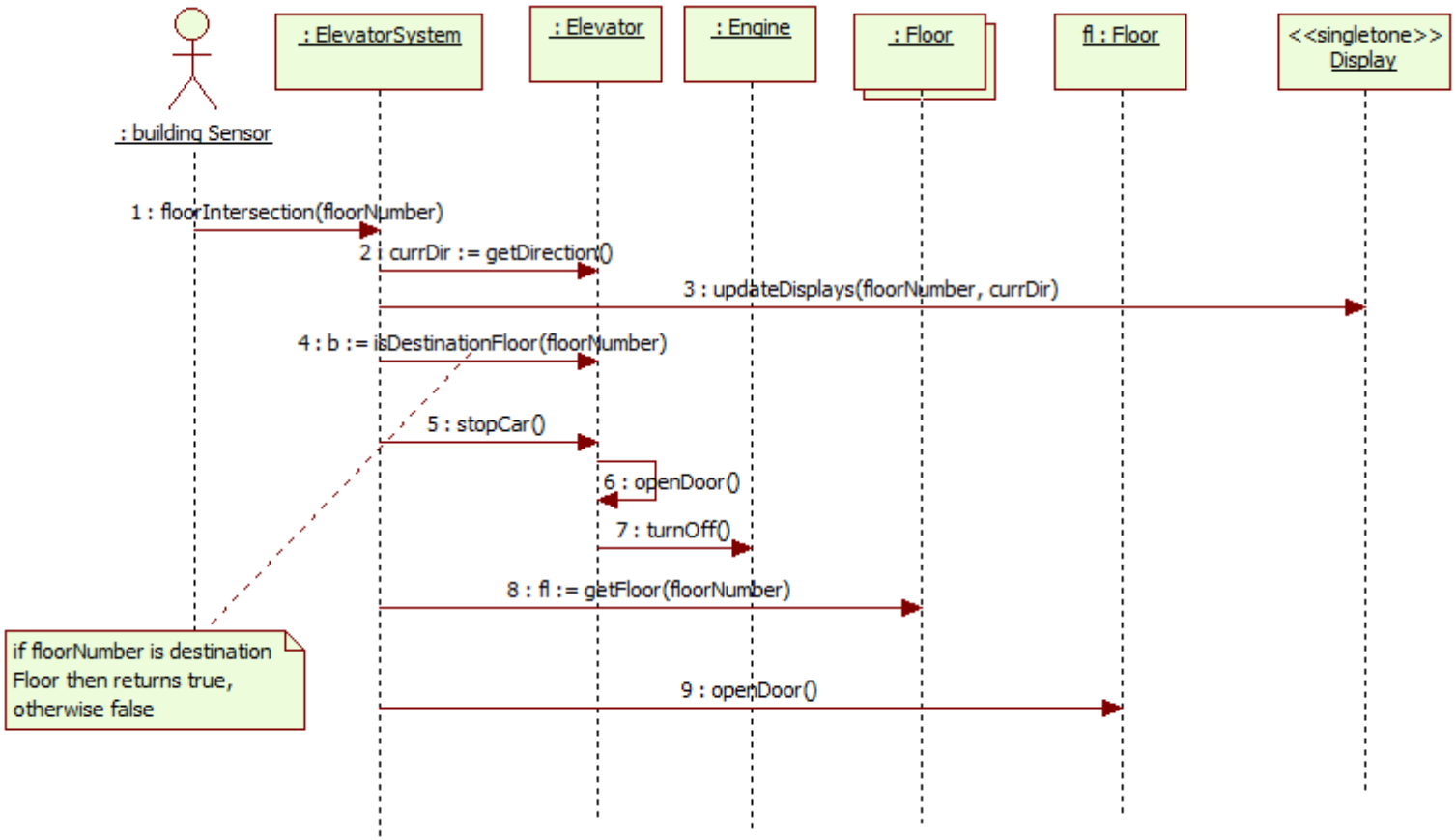
Open Event


: passenger
inCabinButton : Button
: ElevatorSystem
: Elevator
: Floor
fl : Floor
elDoor : Door
fDoor : Door
1 : pressButton()
2 : activateButton()
3 : openDoors()
4 : b := getStatus()
5 [b is Waiting] : currPos := getCurrentFloor()
6 [b is Waiting] : openDoor()
7 : open()
8 : fl := getFloor(currPos)
9 : openDoor()
10 : open()
check If we can open
an elevator or not

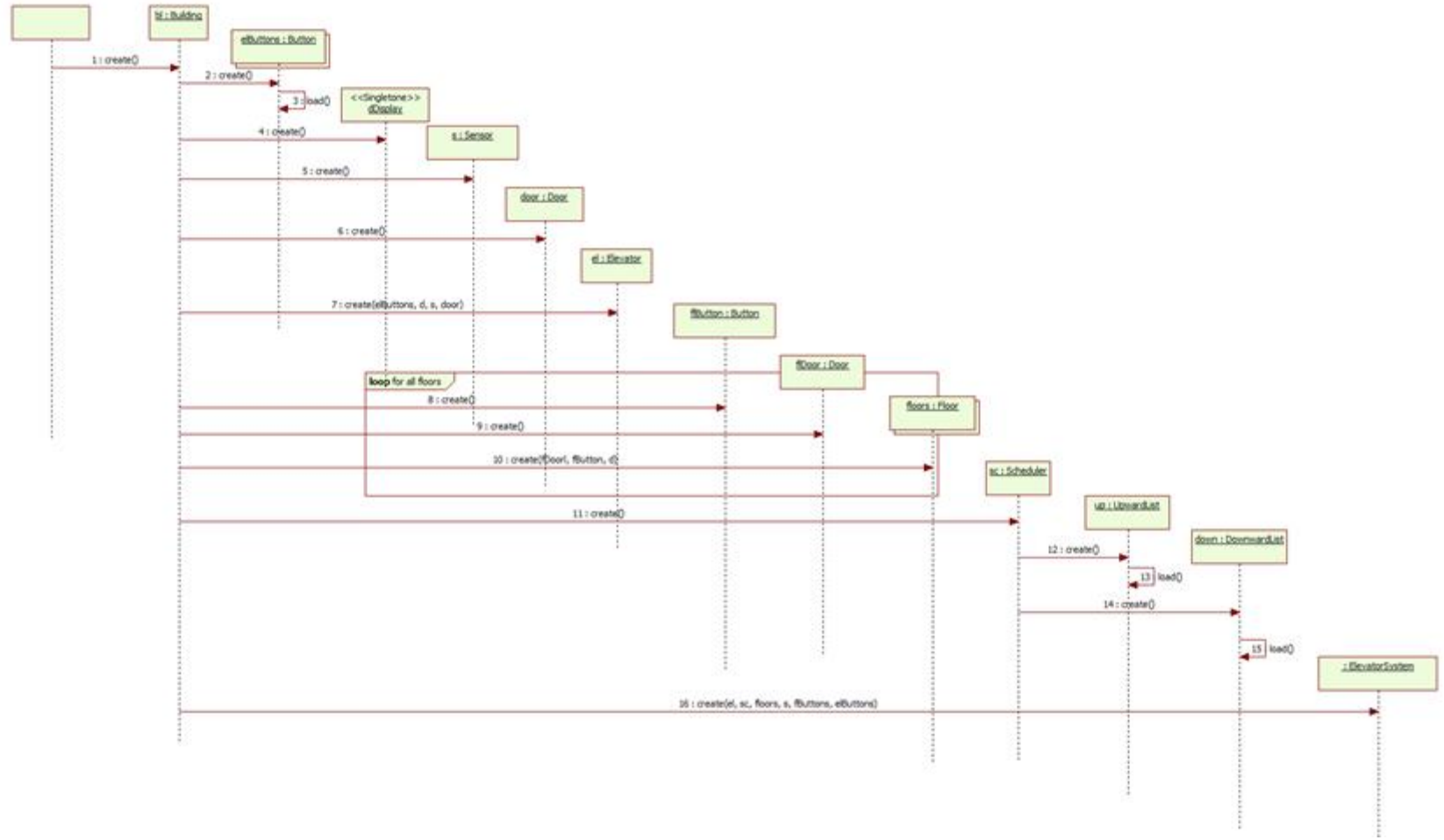
Close Event


: passenger
inCabineButton : Button
: ElevatorSystem
: Elevator
: Floor
fl : Floor
elDoor : Door
flDoor : Door
1 : pressButton()
2 : activateButton()
3 : closeDoors()
4 : currPos := getCurrentFloor()
5 : closeDoor()
6 : close()
7 : fl := getFloor()
8 : closeDoor()
9 : close()

Elevator Intersection



Start Up


bl : Building
elButtons : Button
1 : create()
2 : create()
3 : load()
<<Singletone>>
dDisplay
4 : create()
s : Sensor
5 : create()
door : Door
6 : create()
el : Elevator
7 : create(elButtons, d, s, door)
flButton : Button
fDoor : Door
loop for all floors
8 : create()
floors : Floor
9 : create()
10 : create(fDoor, flButton, d)
sc : Scheduler
11 : create()
up : UpwardList
12 : create()
down : DownwardList
13 : load()
14 : create()
15 : load()
: IElevatorSystem
16 : create(el, sc, floors, s, flButtons, elButtons)

3.Design Model: Class diagram