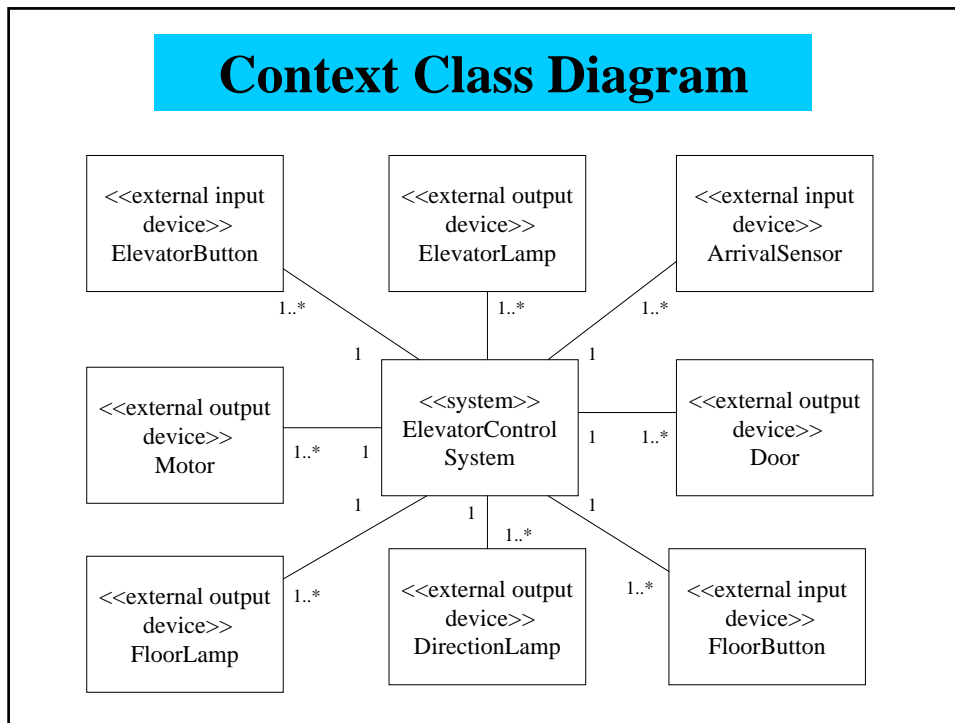


Elevator Control System

Koichiro Ochimizu
School of Information Science
Japan Advanced Institute of Science and Technology

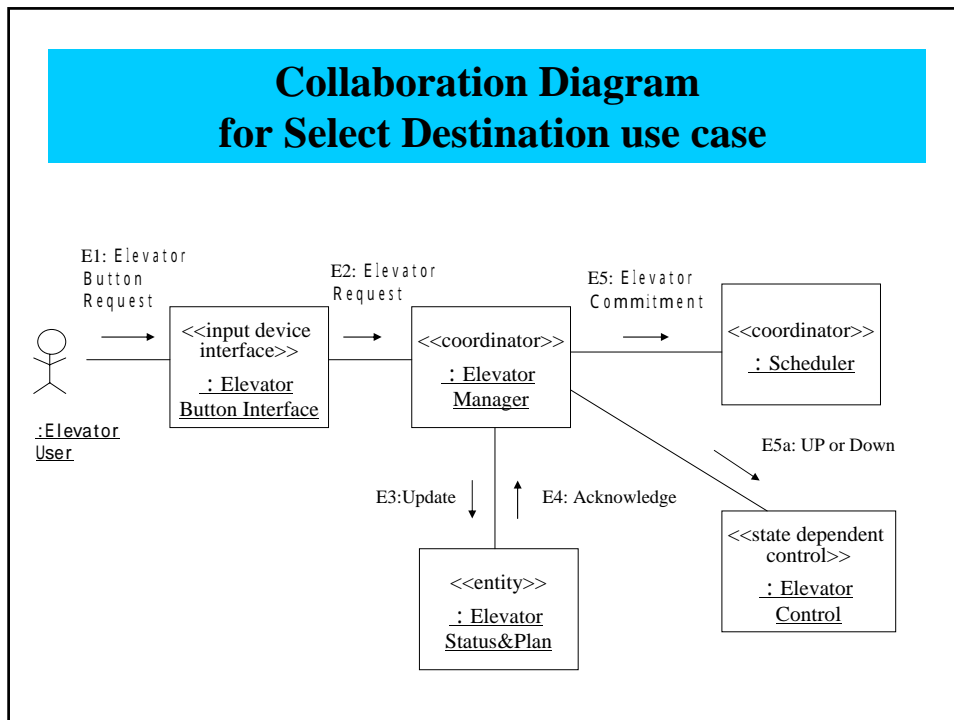
Schedule(3/3)

- March 12
 - 13:00 Unified Process and COMET
 - 14:30 Case Study of Elevator Control System
(problem definition, use case model)
- March 13
 - 13:00 Case Study of Elevator Control System
(finding analysis classes by developing a consolidated communication diagram)
 - 14:30 Case Study of Elevator Control System
(sub-system structuring and task structuring)
- March 14
 - 13:00 Case Study of Elevator Control System
(performance analysis)
 - 14:30 UML2.0 and MDA



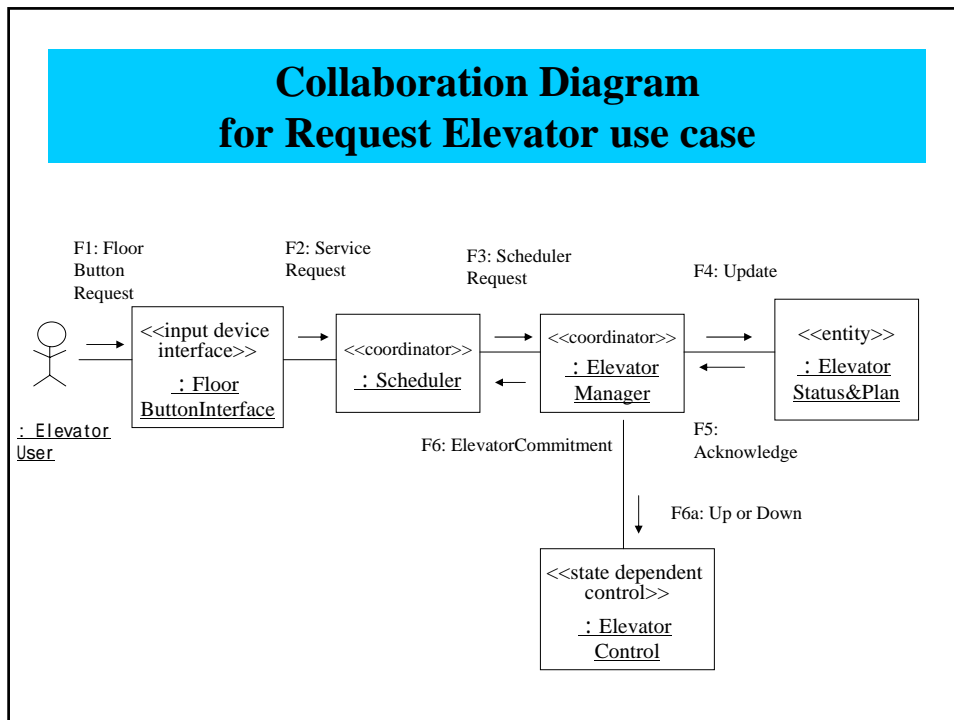
Finding Analysis objects

- For every <<external input/output device>> object, there is a corresponding software device interface object.
- Each elevator has a state-dependent control object called Elevator Control, which control the elevator motor and door.
- Because requests or the elevator can come at any time, a decision is made to have a separate coordinator object, called the Elevator Manager, to receive all incoming requests for the elevator and to update the elevator plan.
- An entity object is needed for each elevator, which we call Elevator Status & Plan.



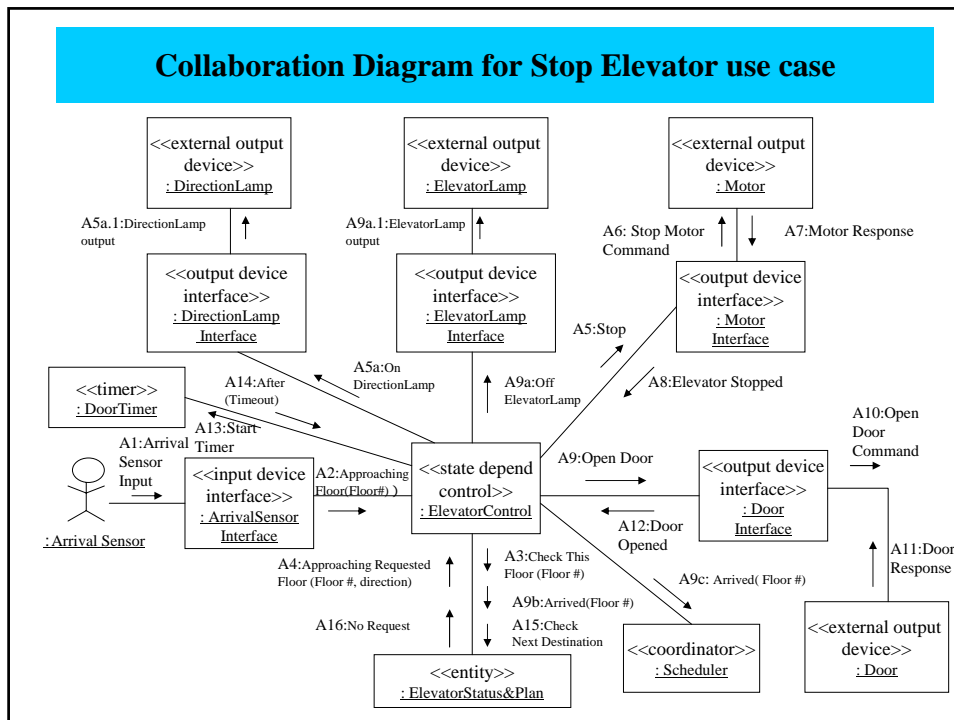
The message sequence description

- E1: The Elevator Button Request arrives at the Elevator Button Interface object.
- E2: The Elevator Button Interface object sends the Elevator Request to the Elevator Manager object.
- E3: The Elevator Manager sends the request to Elevator Status & Plan, which adds the request to the list of floors to be visited.
- E4: The elevator plan is updated. An acknowledgement is returned to the Elevator Manager object, which identifies whether the elevator is idle.
- E5: The Elevator Manager sends an Elevator Commitment message to the Scheduler, to inform it that this elevator is committed to visit the given floor.
- E5a: If the Elevator is idle, the Elevator Manager sends an Up (or Down) message to the Elevator Control object, directing it to move in the desired direction. This case is handled by the Dispatch Elevator use case.



The message sequence description

- F1: The Floor Button Request arrives at the Floor Button Interface object.
- F2: The Floor Button Interface object sends a Service Request to the Scheduler object.
- F3: The Scheduler object selects an elevator and sends a Scheduler Request to the Elevator Manager object in the selected Elevator composite object.
- F4: The Elevator Manager object sends an Update message to the Elevator Status & Plan to add the new request to the elevator plan of which floor it is to visit.
- F5: An acknowledgement is returned to the Elevator Manager object, which identifies whether the elevator is idle.
- F6: The Elevator Manager sends an Elevator Commitment message to the Scheduler.
- F6a: If the elevator is idle, the Elevator Manager sends an Up (or Down) message to the Elevator Control object, directing it to move in the desired direction. This case is handled by the Dispatch Elevator use case.



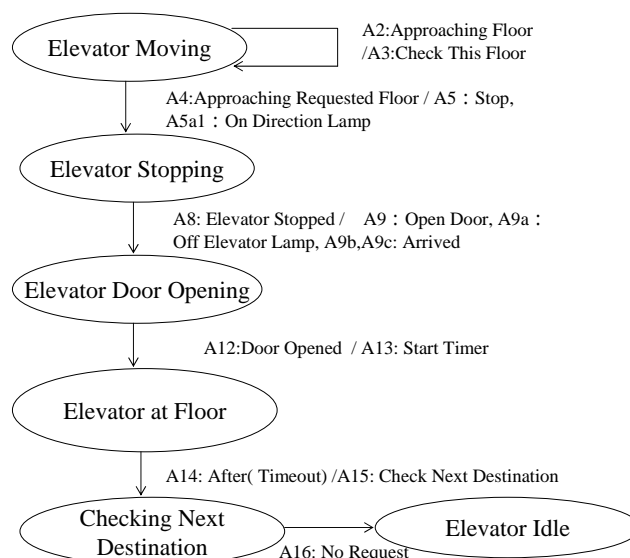
The message sequence description

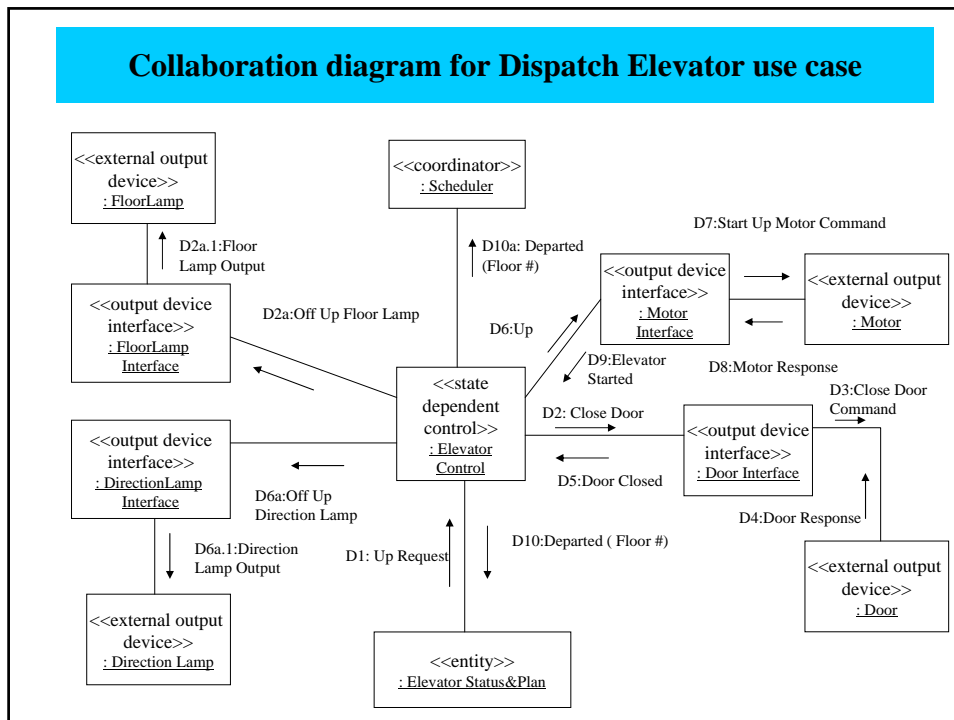
- A1: The Arrival Sensor Interface object receives an input from the arrival sensor external entity.
- A2: The Arrival Sensor Interface object sends the floor number in the Approaching Floor message to the Elevator Control object.
- A3: The Elevator Control object sends a Check This Floor message to the Elevator Status & Plan object, which checks whether the floor at which the elevator is arriving is one where it should stop.
- A4: As the elevator is arriving at a requested floor, the Elevator Status & Plan object sends the Approaching Requested Floor message to the Elevator Control object. The message contains the floor number and the future direction. On receiving this message, Elevator Control transitions from Elevator Moving state to Elevator Stopping state.
- A5: As a result of the transition to Elevator Stopping state, the Elevator Control object commands the Motor Interface object to Stop.
- A5a(parallel sequence): Elevator Control sends an On Direction Lamp (with up or down as a parameter) to the Direction Lamp Interface object, which switches on the real-world direction lamp(A5a.1).
- A6: The Motor Interface object sends the Stop Motor Command to the real-world motor.
- A7: The Motor Interface object receives the Motor Response.
- A8: Motor Interface object sends an Elevator Stopped message to the Elevator Control object, which then transitions to Elevator Door opening state.

The message sequence description

- A9: On transitioning to Elevator Door Opening state, the Elevator Control object sends the Door Interface object a command to Open Door.
- A9a(parallel sequence because there are four actions associated with the state transition): The Elevator Control object sends an Off Elevator Lamp message to the Elevator Lamp Interface object, which then sends an Elevator Lamp Output to the external lamp to switch it off(A9a.1).The Elevator Control object sends the Arrived message to both the Elevator Status & Plan object(A9b, third parallel sequence) and the Scheduler object(A9c, Fourth parallel sequence).
- A10: The Door Interface object sends the Open Door Command to the real world door.
- A11: The Door Interface object receives the Door Response.
- A12: The Door Interface object sends a Door Opened message to the Elevator Control object, which then transitions to Elevator at Floor
- A13: The Elevator Control object starts a timer.
- A14: A timer event is generated after a period of time equal to timeout, causing the Elevator Control object to transition to Checking Next Destination state..
- A15: As a result of the transition, Elevator Control sends a Check Next Destination message to the Elevator Status & Plan object. The objective is to determine the next destination just prior to departure, in case there has been a recent update to the plan. If the elevator does not have any outstanding requests, it transitions to Elevator Idle state (event A16). Otherwise, use the Dispatch Elevator use

Stop Elevator at Floor use case: statechart for Elevator Control





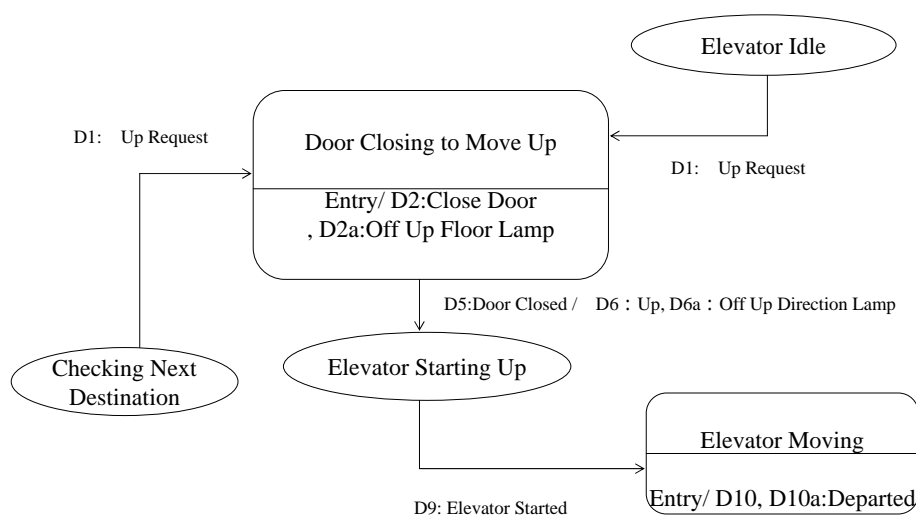
The message sequence description

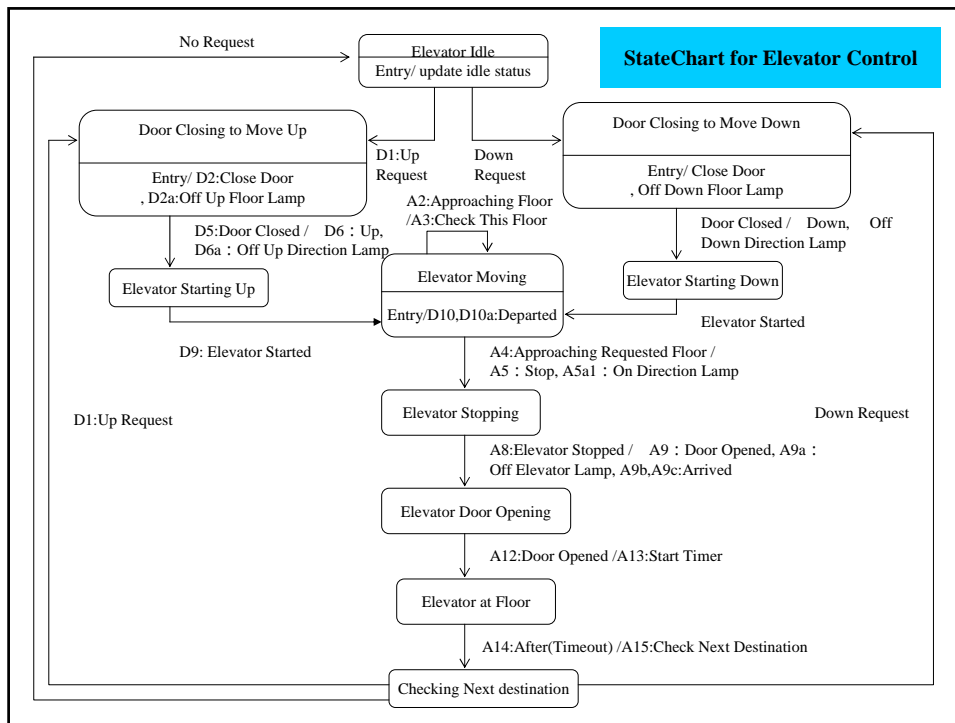
- Starting preconditions are different for Dispatch Elevator.
 - Stop Elevator at Floor is the first case. On entering Checking Next Destination state, Elevator Control sends a Check Next Destination message to Elevator Status & Plan. Elevator Status & Plan sends an Up Request (or Down Request) message to Elevator Control, informing it of the direction in which to move.
 - Elevator Control object is in Elevator Idle state is the second case. Elevator Manager receives a message from either the Scheduler or the Elevator Button Interface with a request for the elevator to visit the floor. Elevator Manager sends a message to Elevator Status & Plan to update the plan. If the elevator is busy servicing a request, Elevator Status & Plan returns an Acknowledgement message with a null parameter. On the other hand, if the elevator is idle, Elevator Status & Plan returns an Acknowledgement message with an up (or down) parameter.
- D1: {Source object} sends Elevator Control an Up Request message. Elevator Control transitions to Door Closing to Move Up state.
- D2: As a result of this state transition, there are two concurrent outputs events. Elevator Control sends a Close Door command to Door Interface. On the statechart, the Close door event (as well as one other output event) is shown as an entry action, because the Up Request event can arrive from either the Elevator Idle state or the Checking Next Destination state.
- D2a(parallel sequence): Elevator Control sends an Off Up Floor Lamp to the Floor Lamp Interface object, which switches off the real-world

The message sequence description

- D3: Door Interface sends a Close Door Command to the real – world door.
- D4: The real-world door sends a Door Response when the door is closed.
- D5: The Door Interface sends a Door Closed message to Elevator Control, which transitions to Elevator Starting Up state.
- D6: Elevator Control sends an Up Command to the Motor Interface object.
- D6a: Elevator Control sends an Off Up Direction Lamp request to the Direction Lamp Interface object, which switches off the direction lamp.
- D7: The Motor Interface object sends the Start Up Motor Command to the real-world motor.
- D8: The real-world motor sends a Motor Response when the elevator has started moving upward.
- D9: The Motor Interface object sends an Elevator Started message to Elevator Control, which transitions to Elevator Moving state.
- D10: Elevator Control sends a Departed message to both the Elevator Status & Plan and Scheduler object.

Dispatch Elevator use case: Statechart for Elevator Control





Consolidated Collaboration Diagram

- Consolidated Collaboration Diagram shows all the objects that participate In the use cases and all the interactions between these objects.

