Tutorial: specify the lifecycle of different parts of the elevator controller

Jérémie Tatibouët, Shuai Li, Patrick Tessier, Asma Smaoui and François Terrier

Lab 1: active classes

RequestProcessor, **MotorControl** and **EntranceControl** are active classes. This means their lifecycle (i.e., the way they evolve over time) needs to be defined as classifier behaviors.

1. Set these classes as active.
2. Attach a classifier behavior (a state-machine) to each of these classes.

Lab 2: state-machine of the class "MotorControl"

Signals, which are used to communicate asynchronously between the different parts, should be defined in the package "**Signals**".

The state-machine of the class "**MotorControl**" must be defined. It must meets the following requirements:

- The elevator car is either "stopped" or "moving".
- The action of moving the elevator car is controlled by the arrival of a request emitted by the request processor.
- The action of stopping the elevator car is controlled by the arrival of request emitted from the environment (i.e., an element that is external to the controller) to the motor control component.
- When the elevator car gets stopped it notifies the request processor.
- It is possible to update the destination at which the elevator car must stop while it is moving (typically if the travel plan was modified).

Lab 3: state-machine of the class "EntranceControl"

The state-machine of the class "**EntranceControl**" must be defined. It must meets the following requirements:

- While the elevator is moving the door remain closed

- The door opens upon the arrival of a request emitted from the request processor.
- The door remains opened 5 seconds after the last passenger gets in or leaves the elevator car.
- When the door is finally closed the request processor is notified.
- **Bonus**: you can integrate a control of the weight limit for the elevator car. If you do so add the required ports to get incoming values provided by the weight sensor.

Lab 4: state-machine of the class “RequestProcessor”

The state-machine of the class “**RequestProcessor**” must be defined. It must meets the following requirements:

- The “RequestProcessor” stays idle while its travel plan is empty.
- As soon as its travel plan is updated the “RequestProcessor” starts to process it.
- Processing a request consists in:
  1. Moving the elevator at the required floor (only if current floor different from the destination floor)
  2. Controlling the door opening and closing when arrived at the destination floor.
- While processing a request, the travel plan might be modified (e.g., stop required before the currently defined destination floor).