MODEL-BASED DEVELOPMENT - LAB

Objectives

- To ensure your understanding of the lecture series, and tutorial.
  - You will further build on the dishwasher model from the tutorial.
  - You will develop a simple model starting from some requirements for a simple train door controller.

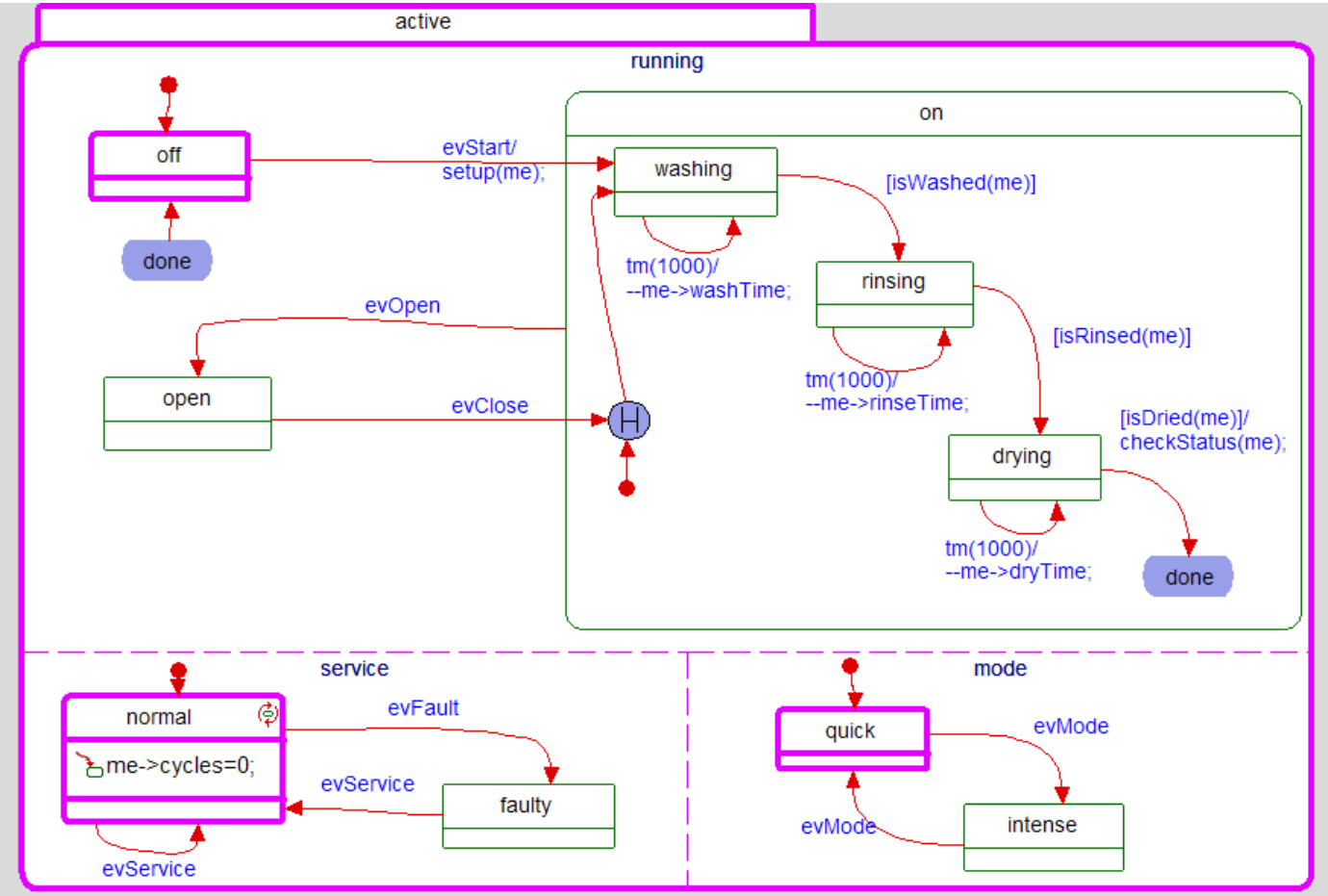
Make sure you have completed your tutorial before proceeding with this lab exercise.

Instructions

- For approval, you are expected to correctly answer and demonstrate all of the questions in this exercise.
- You are expected to answer & demonstrate orally.
- You are expected to answer & demonstrate in a single event once you have completed all of the questions.
  - Prepare your answers
  - You may need to save different versions of the project, in order to demonstrate different questions.

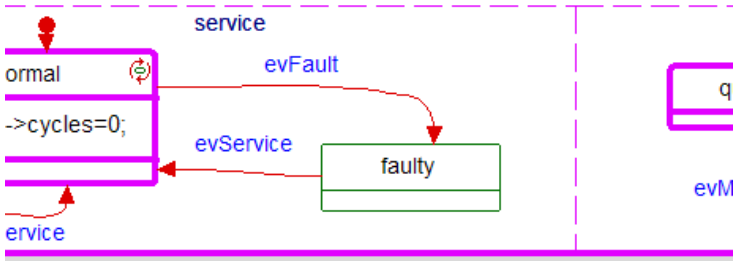
Part1: The dishwasher

- We assume that you have a ready model of the dishwasher from the earlier tutorial.



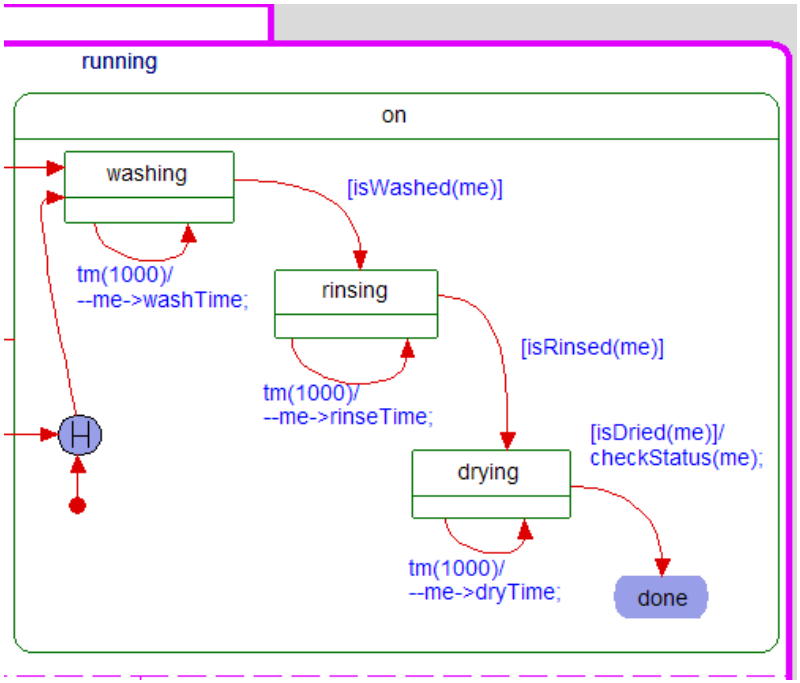
Question1

- When does the dishwasher end up in the state *faulty*?
  - Demonstrate by animating the sequence of events needed to reach that state from the initial execution of the model.
- What needs to be done for the system to leave the *faulty* state?
  - Demonstrate



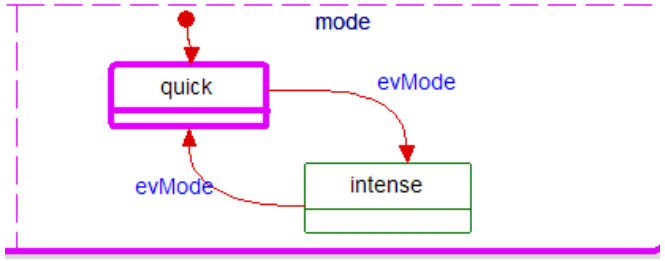
Question2

- What causes the system to move from the *washing* to the *rinsing* state?
  - Explain by highlighting exactly where in the code is the different logic implemented.
- Modify your system design so that the rinsing time is doubled.
  - Where in the code did you perform the modification?



Question3

- Currently, the system contains 2 modes of operation: *quick* & *intense*. Modify the system so that it toggles between the following 3 modes of operation: *quick*, *Intensive*, & *energySaver*.
  - It is up to the designer to decide on the order of operation. However, the system should rotate between these modes indefinitely.



Question4

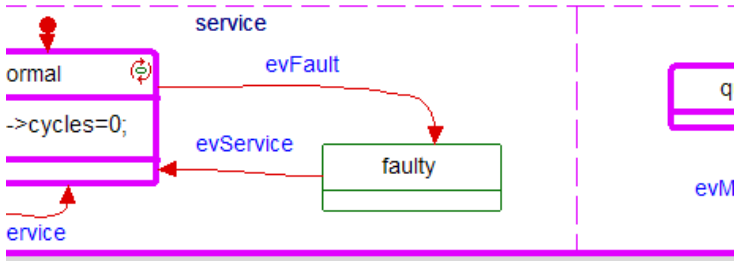
- Describe the current behaviour of the system when the user opens the machine (through the evOpen event) and then closes it (through the evClose event).
  - Does the behaviour differ depending on when the user opens the machine? Explain!
- Modify the behaviour so that the machine always starts its program from the *washing* state when the user closes again.

Question5

- Consider the scenario when the user generates the *evStart* and the system is in the *on* state.
  - What happens if the user now generates another *evStart* event? Why?
  - Demonstrate that your explanation matches the model animation!

Question6

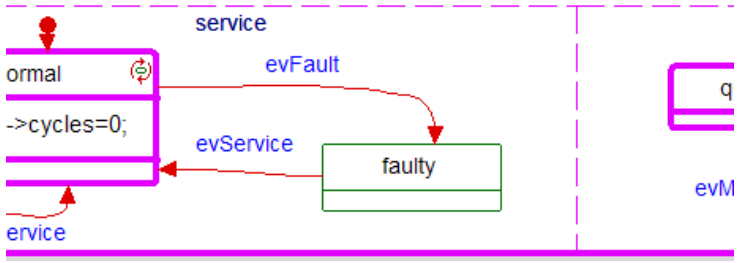
- Currently, the dishwasher works as usual even if it is faulty.
  - Modify the system so that the dishwasher is prevented from being started, once it is faulty.
  - Modify the system so that the dishwasher is also turned off, once it is faulty.



Hint: `IS_IN` is a macro that tests to see if the object is in a particular state. (You could use the IS_IN macro also in a guard)

Question7

- Currently, the user can change the mode of operation at any time.
  - Modify the system so that the user cannot change the mode if the machine is turned on.

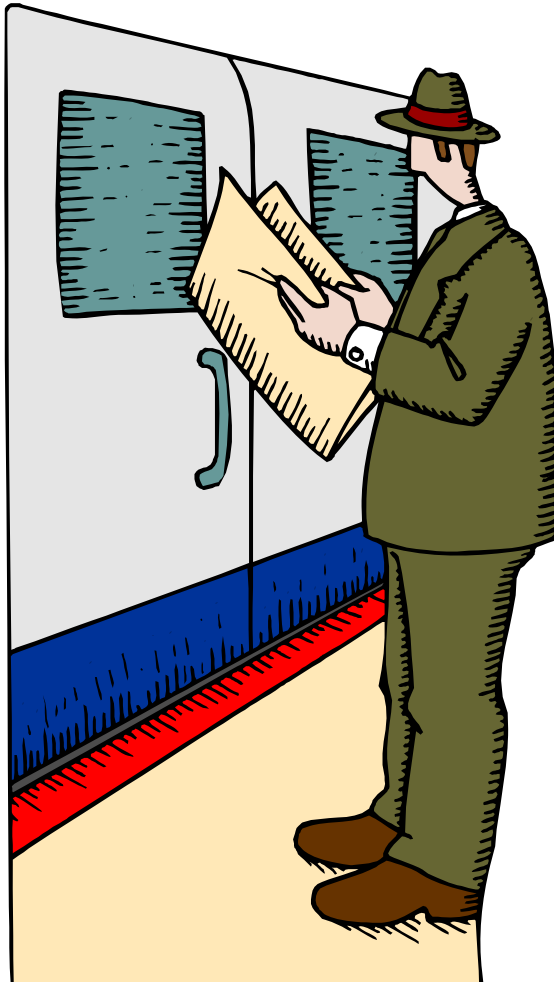


Question8

- While in the *drying* state (but no other state), allow the user to be able to interrupt the washing, and set the machine to off!
  - Demonstrate!

Part2: A train door controller

- You are to develop a state machine model of a train door controller based on some given specifications.
  - You will first focus on a subset of the system requirements that deal with a stationary train.
  - You will then need to modify your model to handle additional requirements that deal with the train in motion.



Door controller requirements

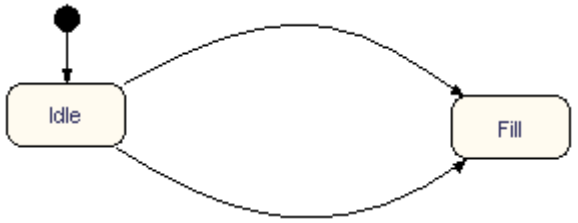
Using a state machine, design the software logic of a train door controller that has the following requirements while the train is stationed at the platform:

- The door controller is to ensure the safe opening and closing of a train door based on input from passengers, train operators, and its sensors.
- If the door is closed, the door is to be opened when the user presses the open button.
- The door is to be closed 10 seconds after it has been opened.
- If the system detects a passenger entering while the door is being closed, the door is to be opened again.
- If the a passenger presses the open button while he door is being closed, the door is to be opened again.

Tips: State machine design guidelines

- Identify and define states of the System

- Identify and define transitions



- Identify and define events and actions

Additional door controller requirements

Now that you have a basic model working, modify the earlier system to also handle the following needs when the train is no longer at a platform:

- The system should never respond to the open button while the train is no longer at a station.
- The operator can enable/disable the door opening button. (naturally, when the train is not at a platform, the operator cannot enable the door opening button. If such an attempt is made, a warning should be given to the operator).

- A train is defined to be a platform if it has had a speed of 0 km/hr for at least 3 seconds

Question1: A train door controller

- In your final model, did you identify any additional behaviour not earlier specified in the textual requirements?
  - If so, describe!