Prof. Dr. A. Podelski, Dr. B. Westphal
C. Schätzle

Wintersemester 2016/17

Software Design, Modeling, and Analysis in UML

http://swt.informatik.uni-freiburg.de/teaching/WS2016-17/sdmauml

Exercise Sheet 5

Early submission: Monday, 2016-12-19, 12:00 Regular submission: Tuesday, 2016-12-20, 8:00

Exercise 1 (3/20 Points)

The class diagram in Figure 1 induces a signature $\mathscr{S}_0$.

Provide the corresponding signature $\mathscr{S}$ according to the definition from the lecture.

Exercise 2 — Ether (3/20 Points)

Consider the FIFO queue ether

$$\varepsilon = \underline{(u_2, e_1)}.(u_1, e_2)$$

with respect to the system state defined by the complete object diagram in Figure 2. (First entry of the FIFO underlined for clarity.)

(i) Compute $ready(\varepsilon, u_1)$ and $ready(\varepsilon, u_2)$. (1)

(ii) Which ether do we obtain from

$$\oplus(\ominus(\ominus(\varepsilon, e_1), e_2), u_1, e_3)$$

assuming that e_3 is the identity of an instance of signal WQ?

Provide the intermediate results to support your claim. (2)

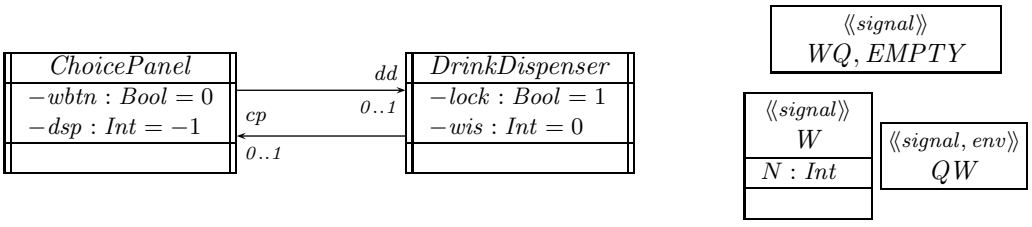


Figure 1: Vending machine class diagram.

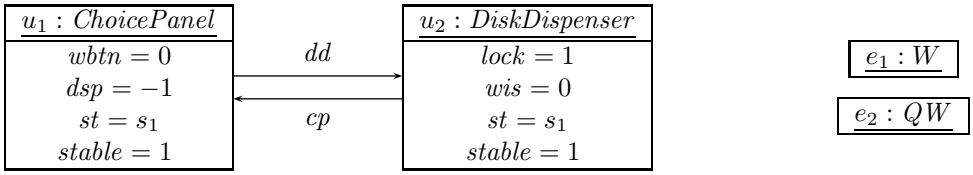
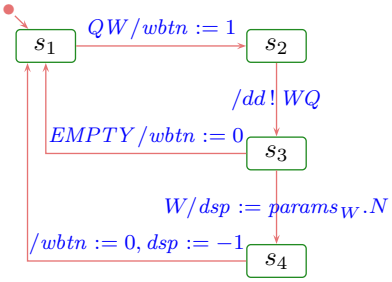
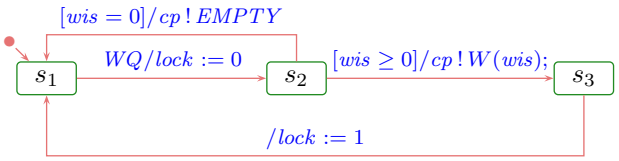


Figure 2: Complete object diagram.



(a) Statemachine $\mathcal{SM}_{CP}$ of choice panels.



(b) Statemachine $\mathcal{SM}_{DD}$ of drink dispensers.

Figure 3: State machines.

Exercise 3 (3/20 Points)

Consider the system configuration (σ, ε) where system state σ is a system state (wrt. signature σ from Exercise 2) as defined by the complete object diagram in Figure 2 and ε is the ether from Exercise 1. What are the possible effects on (σ, ε) of object u_1 executing

(i) action `skip`? (1)

(ii) action $wbtn := 1$? (1)

(iii) action $dd\,!\,WQ$? (1)

Exercise 4 (11/20 Points)

Consider the system configuration (σ, ε) where system state σ is defined by the complete object diagram in Figure 2 and ε is the ether from Exercise 1

(i) How can the model including the state machines given in Figure 3 evolve from (σ, ε) (when not using the rule for environment interaction)?

(For each step, note down the rule by which it is justified.) (5)

Hint: choose a convenient and readable representation of the possible steps, e.g., a table similar to the one in the lecture.

(ii) Explain in your own words the difference between step and RTC-step using examples from your results for the previous task. (1)

Hint: how are step and RTC-step related? Can you point out a step which is also an RTC-step? An RTC-step which is (or is not) a step?

(iii) Is it possible to reach the designated error configuration from (σ, ε)? If yes, point out how – if not, propose a modification of the state machines or of the considered system configuration such that the error configuration is reached. (1)

(iv) If we also consider the rule for environment interaction, how does the possible behaviour change? (1)

(v) Use the Rhapsody model from the previous exercise sheet to generate code and simulate the model.

How does the behaviour that you can simulate with Rhapsody relate to your results from Task (i)? Is all the behaviour you claimed there available in Rhapsody? Is it more, is it less? Why? (3)

Document the behaviour observed with Rhapsody simulation by submitting your model including (a) recorded sequence diagram(s).

Exercise 5 (10 Bonus)

(i) The idea of the choice panel behaviour model in Figure 3 is that the choice panel displays the amount of water units in stock *right before dispensing* the water which is just dispensed. We assume that attribute *dsp* is wired to a physical display on the actual machine which shows non-negative numbers, and which is off if *dsp* is negative. (Maybe somebody wants to inform the maintenance personnel if the number drops too low.)

There is a requirement on the vending machine that whenever a choice panel is in state s_4 (corresponding to just dispensing a drink), the display shows a positive number.

This requirement has been formalised with the following OCL constraint

$$\mathsf{context}\ CP\ \mathsf{inv} : st = s_4\ \mathsf{implies}\ dd.wis > 0$$

Is this constraint an invariant of the behaviour which is possible from system configuration (σ, ε)?

If yes, argue why, if no, point out a counter-example. (5 Bonus)

(ii) If you found the constraint from the previous task not to be an invariant of the model, propose a change to the state machines such that the constraint becomes an invariant (and the overall model still behaves sufficiently similar to a reasonable vending machine).

To prove that a constraint *is not* an invariant of the behaviour is easy: just provide a counter-example.

But how can we *prove* that an OCL constraint is an invariant of the (possibly changed) model?

(4 Bonus)

(iii) Is the problem whether a given OCL constraint is an invariant of the behaviour of a given UML model from a given system configuration decidable in general? (1 Bonus)