Prof. Dr. A. Podelski, Dr. B. Westphal
E. Plaku Wintersemester 2011/12

Software Design, Modeling, and Analysis in UML

`http://swt.informatik.uni-freiburg.de/teaching/winter-term-2011-2012/sdmauml/sdmauml`

Exercise Sheet 4

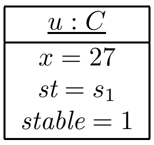
Early submission: Monday, 2012-01-09, 12:00 Regular submission: Tuesday, 2012-01-10, 12:00

Exercise 1 (12/20 Points)

Assume, the state machine shown in Figure 1 is the state machine of class C.

- Provide a class diagram $\mathcal{CD}$ such that Figure 1 is a well-formed state machine diagram wrt. $\mathcal{CD}$ and give the core state machine corresponding to Figure 1. (3)

- Let (σ, ε) be the system configuration where σ is given by the following complete object diagram



  and ε is a FIFO which is shared between all alive objects and which comprises exactly the four events E, E, F, G for u in that order (E is first, G is last).

  How can (σ, ε) evolve?

  For each transition $(\sigma, \varepsilon) \xrightarrow[u]{(cons, Snd)} (\sigma', \varepsilon')$ in the evolution, please point out which of the cases (i) – (v) justifies the transition, how are system state σ' and ether ε' determined according to the rule? (7)

- In your answer to the previous task point out an example for a step and an example for an RTC step. (1)

- If we define length of an RTC step to be the number of steps in it, are RTC steps of finite length in general? (1)

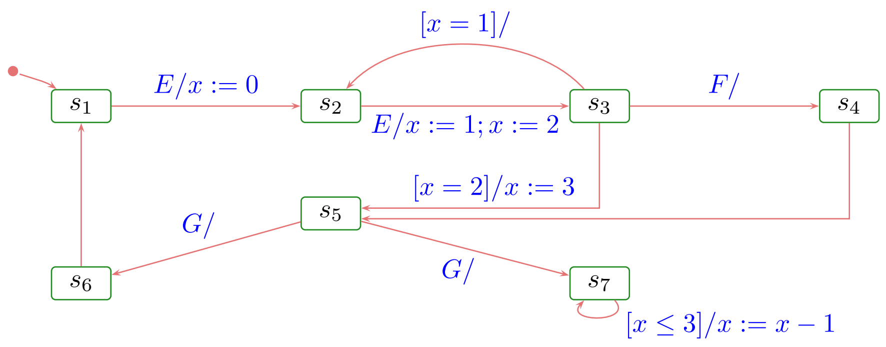


Figure 1: State machine for Exercise 1.

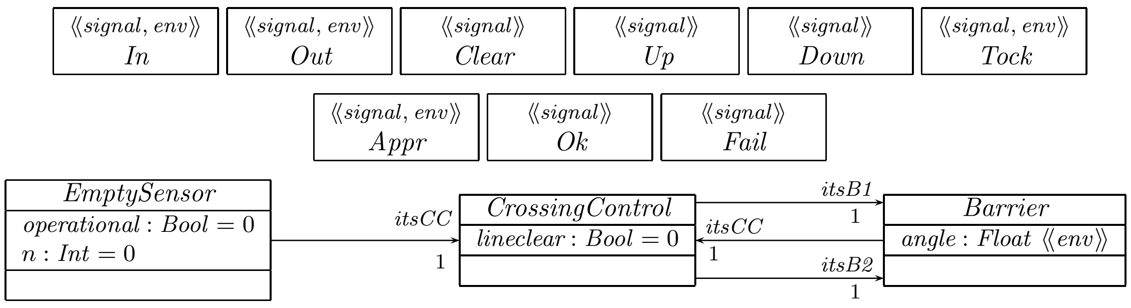
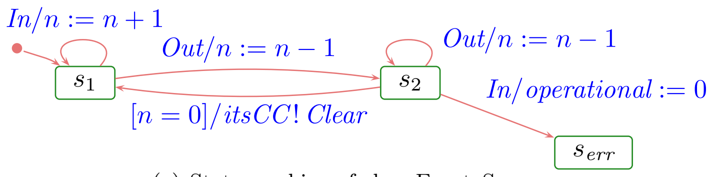
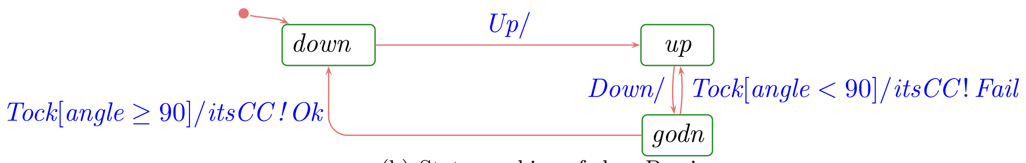


Figure 2: Class diagram for Exercise 2.



(a) State machine of class EmptySensor.



(b) State machine of class Barrier.

Figure 3: State machines for Exercise 2.

Exercise 2 (8/20 Points)

Consider the UML model of a one-way level crossing consisting of Figures 2 and 3. Assume the model already comprises an initialisation phase which ensures that all links are established correctly and that the barriers are all up.
Please provide a state machine for the CrossingController which, for each approaching train first lowers the barriers, then in case the barriers do not hit an obstacle sets *lineclear* to *true* to signal the train that it's safe to proceed, and after the train is through, first resets *lineclear* and then raises the barriers.
Your state machine shall in particular ensure that the line is never cleared if the barriers are not down.
Convince tutor that your design is correct.

Note: the EmptySensor is counting axes. Event 'In' corresponds to a train axis being detected before the crossing and 'Out' to one axis being detected after the crossing.
Hint: Where is the train? Do you need to make any assumptions to ensure that the requirements are satisfied?