Prof. Dr. A. Podelski, Dr. B. Westphal
E. Plaku

Wintersemester 2012/13

Software Design, Modeling, and Analysis in UML

http://swt.informatik.uni-freiburg.de/teaching/WS2012-13/sdmauml

Exercise Sheet 5

Early submission: Monday, 2013-01-14, 12:00 Regular submission: Tuesday, 2013-01-15, 10:00

Exercise 1 (7/20 Points)

Consider the hierarchical state machine shown in Figure 1.

(i) Provide the mathematical representation of Figure 1. (1)

(ii) Point out an example of a basic, composite, OR-, AND-, initial, and final state. (1)

(iii) What *may* happen when dispatching E in the configuration uniquely determined by $\{s_1, s_5\}$? (2)

(iv) What if "first E then F"? (2)

(v) Can "first E, then G" kill the executing object? (1)

Note that this task has two equally important aspects: coming up with the answers and presenting the answers. We've used a couple of different forms to note down what a state-machine does in the lecture and considered different levels of detail. With all lectures on state machines as background, which one do you find most appropriate for this task?
Further note that by the context of the lecture, you're now well after the state of "educated guess" but are able to give exact *and* complete *answers and justify their correctness. Please do so. Explain in particular why the things you claim may happen and why no other.*

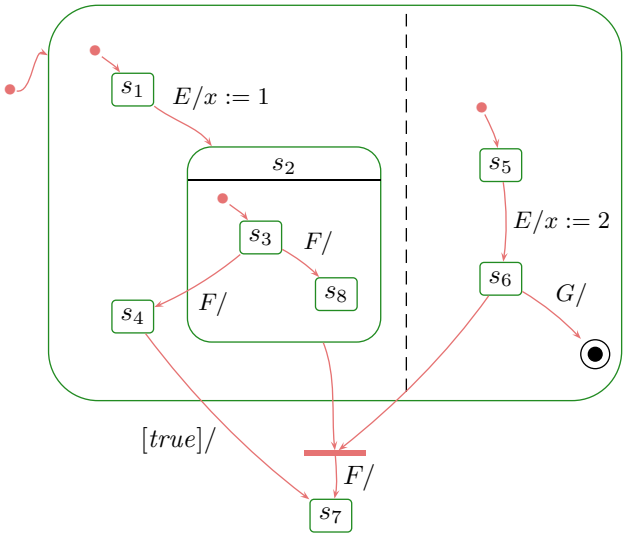


Figure 1: Hierarchical State Machine.

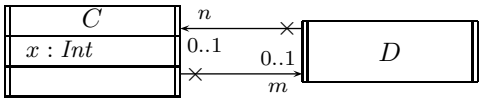
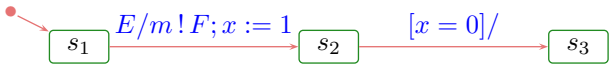
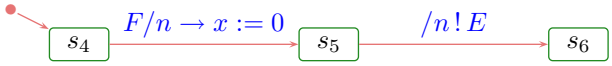


Figure 2: Class diagram for Exercise 3.



(a) State machine of C.



(b) State machine of D.

Figure 3: State machine diagrams for Exercise 3.

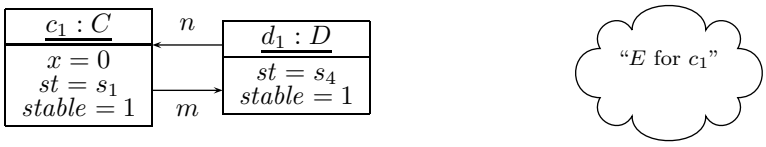


Figure 4: System configuration for Exercise 3.

Exercise 2 (3/20 Points + 5 Bonus)

Consider the UML model given by Figure 2 and 3.

(i) How can the system configuration given by Figure 4 evolve according to the lecture? (1)

(ii) How can the system configuration given by Figure 4 evolve in Rhapsody?
Please provide a sequence diagram.
Discuss. (2)

Bonus: Consider the Statemachine from the last exercise sheet as given by Figure 5. Use Rhapsody to re-do Exercise 5.4: ejecting the four events E, E, F, G in this order, how does the Statemachine evolve?

What happens to the non-determinism in s_5 in Rhapsody? Explain! (5)
Hints: The generated C++ code does not bite.

Exercise 3 (10/20 Points + 5 Bonus)

Recall the level crossing from the last exercise sheet. We have refined the design insofar as that we've added two signals for communication between EmptySensor and your controller. The EmptySensor sends *SensorFail* to its crossing controller before going to state s_{err} when finding a mismatch in number of incoming/outgoing axes. Furthermore, we assume that barriers send their identity with *Ok* and *Fail* signals.
In addition to the last exercise sheet:

- The crossing controller shall operate (to be added) traffic lights.
- The crossing controller shall explicitly wait for *Ok* from its both barriers. The barriers shall be lowered at the same point in time (for efficiency reasons).

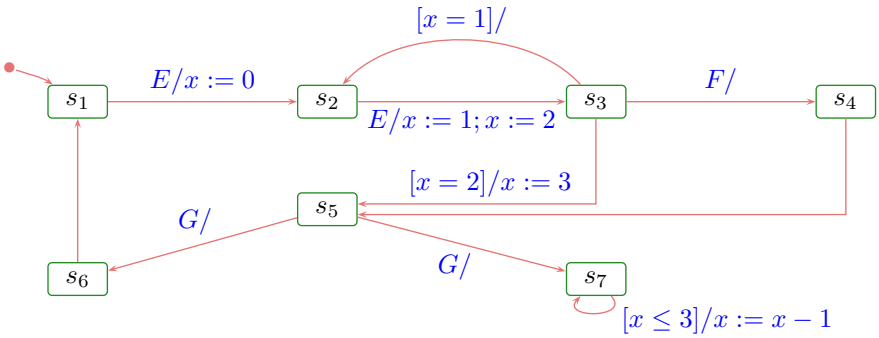


Figure 5: State machine for Exercise 1.

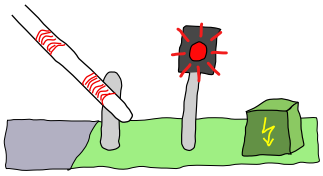


Figure 6: Colourful drawing for aesthetic purposes.

- In case of sensor failure or two failed attempts to close the barriers, the crossing controller shall assume an error state and switch an additional (to be added) far away signal such that far away trains are kept from approaching the crossing.
- Pragmatically, the error state requires some manual interaction where, for instance, obstacles are removed from the track and a train close to the crossing is safely guided over the crossing.

  Completion of the manual interaction shall be signalled to the crossing controller by an additional signal on which the crossing controller shall reset the sensor and the barriers (in a reasonable order).
- In addition to the safety requirement from the last exercise sheet, your proposal shall ensure that, if there are no obstacles in the way and if the sensor is not "confused", then a train will finally get through.

  Bonus: can you formalise the liveness requirement including the assumption in form of LTL over OCL (that is, LTL using OCL expressions instead of atomic propositions)? If you don't know LTL, at least provide the necessary OCL expressions. If necessary, you may propose augmentation of the model.

Please provide a Rhapsody model of crossing control system including your state machine for class *CrossingControl*. Convince your tutor that your proposal is not "completely broken".

Note: in the next exercise sheets, you'll be asked to inspect and "peer review" models of your colleagues. So this time you may want to be particularly "nice to your readers".

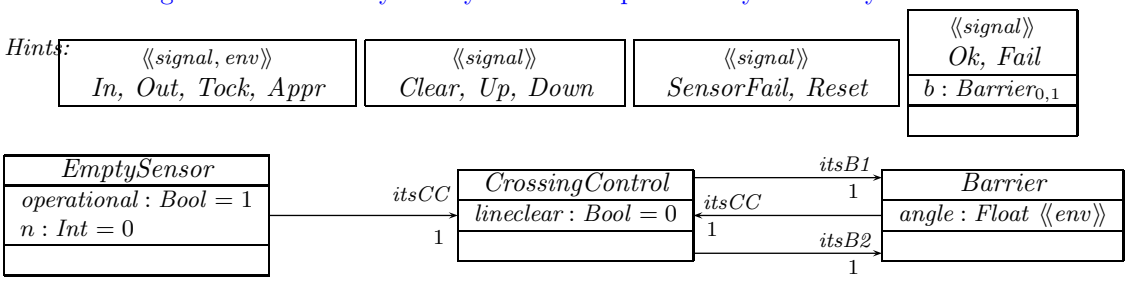
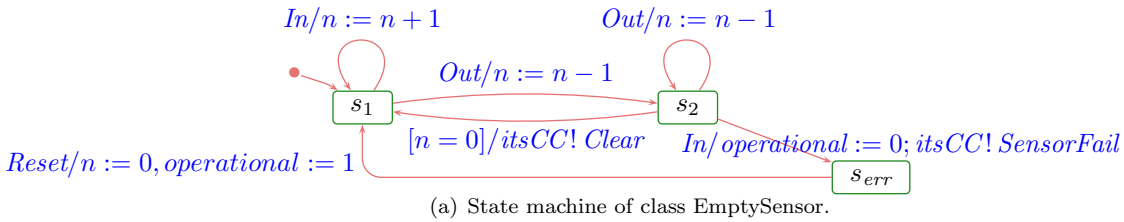
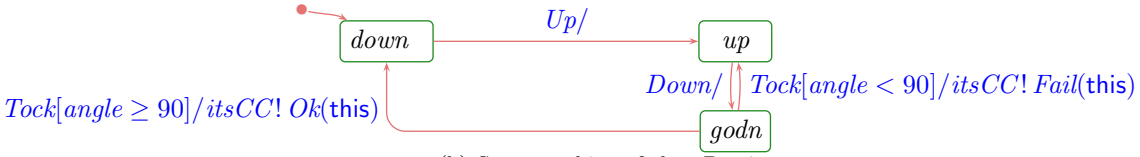


Figure 7: Class diagram for Exercise 2.



(a) State machine of class EmptySensor.



(b) State machine of class Barrier.

Figure 8: State machines for Exercise 2.

- *Recall: we had a brief demo of basic Rhapsody usage in the lecture.*
- *To use Rhapsody, you want to connect to* `architor.informatik.uni-freiburg.de` *with some RDP client.*
- *The host can (for limited number of licences) only run a limited number of parallel instances of Rhapsody. If you don't get a license, please try again later. If the problem persists, tell me.*

  For technical reasons (related with the TCP/IP based communication between compiled application and Rhapsody tool), there can only be a limited amount of animation sessions at the same time. In case of problems, proceed as in the previous case.
- *A Rhapsody model which does not build is worth 0 points. Please ask (first your colleagues and then me) in case you get stuck.*
- *Recall: save your model on a local drive of architor (Desktop, C:, D:), not on a network mounted one. The latter can lead to (strange) build problems.*
- *A good way to convince people of "not completely broken"-ness is to describe the own understanding of the requirements, describe the own design ideas, and provide animated sequence diagrams which show that the model is working as intended.*

  If you do provide sequence diagrams, please explain to your tutor why the particular sequence diagram(s) you submitted are convincing: what do they show?