Prof. Dr. A. Podelski,
Dr. B. Westphal
M. Steinle

Summer term 2020

Softwaretechnik/Software Engineering

http://swt.informatik.uni-freiburg.de/teaching/SS2020/swtvl

Exercise Sheet 3

Early submission: Monday, 2020-06-22, 14:00 Regular submission: Tuesday, 2020-06-23, 14:00

Exercise 1 – Class and Object Diagrams (10/20 Points)

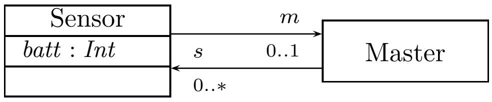

Figure 1: WFAS masters and slaves.

Recall the Wireless Fire Alarm System (WFAS) from Lecture 2. A cornerstone of the design is the idea that for each sensor,[1] there is a so-called master to monitor the sensor and take notifications (like indications low battery). If the system is operational (not in maintenance mode), each sensor must have a link to exactly one master, and the sensor-master links must be reciprocal, i.e. the master's link includes all sensors who have a link to this master.
Figure 1 shows an abstract model of this design idea.

(i) Provide the **abstract syntax** of the diagram. (5)

(ii) Clarify the intended usage of the data-structure as described in the introductory text to this exercise; use at least three *own* (i.e., not from the lecture, not from an exercise) non-trivial system states (over the Class Diagram from Figure 1 and structure $\mathscr{D}$ with $\mathscr{D}(Int) = \mathbb{Z}$ and $\mathscr{D}(\mathscr{C})$ of your choice) shown as Object Diagrams.

- One system state that models a typical configuration in operational mode. (1)
- One system state that models a corner-case configuration, i.e., which may be (dis)allowed by the wording above but which seems questionable from the WFAS context. (1)
- One system state that is not allowed in operational mode. (1)

Hint: Note that the task asks for 'clarification', i.e., a set of system states alone is not a solution to this task. Rather assume that your submission is a review of the proposed structural model as requested by an imaginal 'boss'.

(iii) For one of your Object Diagrams from Task (ii), which includes at least one Sensor-object, provide the underlying system state in function notation. (2)

Exercise 2 – Evaluating OCL Formulae (10/20 Points)

Consider the system state σ_1 given by the complete Object Diagram in Figure 2.

(i) To which value does the Proto-OCL formula

$$F := \forall\, self \in allInstances_{\text{Master}} \bullet \forall\, n \in s(self) \bullet batt(n) > 1$$

evaluate for σ_1? Prove your claim, i.e., compute $\mathcal{I}[\![F]\!](\sigma_1, \emptyset)$. (5)

[1] which keeps track of its battery level

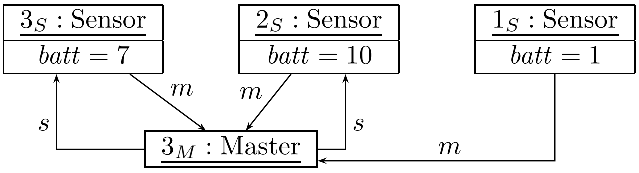

Figure 2: Complete object diagram.

(ii) Provide system states σ_2 and σ_3 such that for each truth value *true*, *false*, or $\perp$, there is an $i \in \{1, 2, 3\}$ such that $\mathcal{I}[\![F]\!](\sigma_i, \emptyset)$ evaluates to this truth value. Argue your claim. (4)

Hint: As we refer to the same OCL formula as in Task (i), you may use the detailed proof that you provided for Task (i) to guide your argument here.

(iii) For which purpose could a software engineer want to compute the evaluation of an OCL formula wrt. an object diagram or create object diagrams for a desired truth value (i.e., conduct tasks like (i) and (ii) above) in a software development project? (1)

Exercise 3 – Writing OCL Formulae (0/20 Points + 3-10 Bonus)

Use Proto-OCL to formalise **ONE OF** the following requirements on the WFAS software:

(i) The battery values of sensors range between 0 and 255. (3 Bonus)

(ii) Each sensor has a master. (5 Bonus)

(iii) Masters must monitor at least one sensor. (5 Bonus)

(iv) If a certain master is master of a sensor (via the m link), then this sensor is monitored by that master (i.e. the sensor is included in this master's s link). (10 Bonus)

Demonstrate that your solution is (at least) plausible:

- Give at least one example of a system state that you expect to satisfy your chosen requirements and at least one that you consider to violate the requirement (i.e., where your formalisation should evaluate to *false*) — together with the outcome you expect.
- Argue why your formula evaluates to the expected outcomes.

Hint: You may use the function symbol '$size : 2^\tau \to \mathbb{Z}_\perp$' with

$$size_\mathcal{I} : \mathcal{I}[\![2^\tau]\!] \to \mathcal{I}[\![\mathbb{Z}_\perp]\!]$$

$$size_\mathcal{I}(x) = \begin{cases} |x| & x \neq \perp \\ \perp & otherwise. \end{cases}$$