Software Engineering
Errata for the Course Slides EN/2018

August 3, 2021

Note: The following ‘repairs’ apply to the slides of 2018 as distributed with the recordings on the ‘EN’ track. Most errors have been fixed with the 2019-slides (except for the one ‘NEW’ one).

Contents

Lecture 4: Software Project Management 1
Slide 35, ‘Building Blocks Can Be Arbitrarily Complicated’ ............... 1

Lecture 9: Scenarios & Use Cases 1
Slide 46, ‘Language of LSC Body: Example’ ............................... 1

Lecture 10: Live Sequence Charts & RE Wrap-Up 1
Slide 8, ‘Loop Condition’ ...................................................... 1
Slide 9, ‘Progress Condition’ (NEW) ...................................... 2
Slide 10, ‘Example’ and Slide 5, ‘Language of LSC Body: Example’ .... 2
Slide 40, ‘LSC Semantics with Pre-Chart’ .................................. 2

Lecture 12: Structural Software Modelling II 2
Slide 28, ‘More Interesting Example’ ....................................... 2

Lecture 4: Software Project Management

Slide 35, ‘Building Blocks Can Be Arbitrarily Complicated’
The first item in the list to the right of the diagram needs to read:

If a test detected an error in $M$,

Lecture 9: Scenarios & Use Cases

Slide 46, ‘Language of LSC Body: Example’
Lecture 10: Live Sequence Charts & RE Wrap-Up

Slide 8, ‘Loop Condition’

The message aspect of the loop condition (first bullet point) needs to read

\[ \psi^{\text{Mag}}(q) = \neg \bigwedge_{1 \leq i \leq n, \psi \in \text{Mag}(q \setminus q)} \psi \wedge (\text{strict} \implies \bigwedge_{\psi \in E^f_i \cap \text{Mag}(L)} \neg \psi) \]

that is, in the non-strict case, the loop accepts all letters where none of the messages of any successor cut is sent or received.

Slide 9, ‘Progress Condition’ (NEW)

The last bullet point needs to read:

\[ \psi_{\text{LocInv}}.\bullet(q, q_i) = \bigwedge_{\lambda=(l, \phi, l', \iota') \in \text{LocInv}, \Theta(\lambda) = \theta, \lambda \bullet \text{-active for } (q, q_i)} \phi \]

Local invariant \((l_0, \iota_0, \phi, l_1, \iota_1)\) is \(\bullet \text{-active for } (q, q_i)\) if and only if
- \(l_0 \in (q_i \setminus q) \wedge \iota_0 = \bullet\), or
- \(l_0 \in q \wedge l_1 \notin q_i\), or
- \(l_1 \in (q_i \setminus q) \wedge \iota_1 = \bullet\).

The fixed definition in particular treats the following case correctly, where an observed sequence of messages \(A!?, C!?, A?\) needs to consider the local invariant together with \(C!?, C!?, A!, A?\):

Slide 10, ‘Example’ and Slide 5, ‘Language of LSC Body: Example’

The loop condition of state \(q_6\) needs to read

\[ \neg (G^{l_2,I_1}_1 \lor G^{l_2,I_1}_1) \]

and the progress condition from \(q_4\) to \(q_6\) needs to read

\[ F^{l_2,I_3}_1 \land G^{l_2,I_1}_1 \land \neg G^{l_2,I_1}_1 \]
Slide 40, ‘LSC Semantics with Pre-Chart’

Each of the four inner table cells had one ‘∧’ too much, and the second lines need to read

\[ \land w^1, \ldots, w^m \in \text{Lang}_{\text{fin}}(\mathcal{B}(PC)) \]

and

\[ \land w^{k+1}, \ldots, w^m \in \text{Lang}_{\text{fin}}(\mathcal{B}(PC)) \]

respectively.

Meaning: the sub-word consisting of the 1st (or \(k+1\)-th) up to \(m\)-th letter of word \(w\) is in the ‘finite’ language of the pre-chart TBA, i.e., we read the pre-chart TBA as a standard Deterministic Finite Automaton (DFA) with the standard DFA-acceptance criterion of reaching an accepting state with consumption of the last letter of the word.

Lecture 12: Structural Software Modelling II

Slide 28, ‘More Interesting Example’

The studied Proto-OCL formula needs to read:

\[ \forall c \in \text{allInstances}_C \bullet x(n(c)) \neq 27 \]