Reflection on module Software Modeling & Specification

By team 6:
Pei-Yin Chao, Bas Groenendaal, John Helmes, Philip Mendels, Rik Wesselink, Mehmet Yalvac

Comments on team 5

Activity Diagram – Lure Pet
The diagram creates the possibility that a pet is interacting with another animal while sleeping. The activities 'interacting with another animal' and 'going to a person' result in the same end state.

Activity Diagram – Play with Pet
The activity 'determine elephant desire' is irrelevant. Because there is no return, the user always has to give the candy when the elephant has done his trick. The diagram is rather one-linear.

Sequence Diagram – Play with Pet
This diagram is not in line with the activity diagram. 'Opt' is not used in the same way as in the other sequence diagram. We interpret the 'opt' in the second diagram as an IF statement. The diagram makes it possible for the person to give his Rolo after it is stolen by the elephant. Statements [A] and [B] conflict: if I decide to present a Rolo to the elephant, then I should have checked whether I have Rolos left. Now these steps are separate, which sounds strange to us.
The same counts for statements [C] and [D]: if the elephants wants to play the game, it should have checked whether it is sick of tired. Otherwise, on what information is the first decision make?

Overall comment
The diagrams are based on "the real world situation" where there is a direct interaction between a person and a pet. Another way of making these diagrams would be by looking at the interactions with, and states of the system. Acting it out reflects where the errors in the program are and how 'straightforward' the program is.

Comments on our own diagrams
Our diagrams were inconsistent with each other. We should have developed the diagrams more in parallel, or revise them, because the design develops during building the diagrams and therefore the diagrams change along the way. Not everything was included in the dynamic diagrams.
It is doubtful whether we should have used 'main menu' and 'interface' as separate classes.

Comments on module
We think the acting out method could be very useful in the beginning of the module, where interpretation, expression play a more important role. Secondly, it would be handy to first act out the logic in the diagram, and after that write the diagram down.
We discovered most flaws by reasoning, but acting out was a good way to communicate flaws in a diagram to the audience.
It was said that the goal of the module was to get a taste of Uml and abstract language, and not to make us experts in Uml. But the focus during the lectures was specifically on details and not on the overall use of those diagrams. That could be a reason why many groups mixed up diagrams.