MTAT.03.083 – Systems Modelling

Regular Exam – 8 January 2013

Notes:

- The exam is open-book and open-laptop. Web browsing is allowed, but you are not allowed to use e-mail clients nor Instant Messaging clients, nor to share any information "live" with anybody inside or outside the exam room.
- Solutions should be submitted electronically using the Submit button on the course web page. Submissions on paper are also accepted, but electronic submissions are preferred.
- For questions 1 and 2, you should submit a MagicDraw project, a Word or a PDF file. For question 2, you should submit a PNML file produced by Woped. Files should be named Question1, Question2 and Question3 respectively.

QUESTION 1: Administration of Meetings **[20 points]**

We consider a simplified version of a system for administrating meetings. The system should maintain a list of users and records of meetings. A meeting has an owner, a list of participants, a time, and a location. The location of a meeting is a meeting room. Each meeting room has a maximum capacity (i.e. maximum number of participants) and is located in a given building and floor. Users may carry out standard operations on meetings, such as creating, reading, editing, and deleting them. A user may also cancel a meeting, which deletes the meeting, frees up the meeting room and notifies all participants by email.

Users are able to administrate meetings through a Web application. When a user successfully logs in into this application, he/she is initially in the state *ListMeetings*. In that state, a user can browse the scheduled meetings and can initiate the editing, creation, deletion and cancellation of meetings. An event of type *edit* causes a transition to the state *editMeeting*, where the currently selected meeting is edited. In this state, the meeting can be saved, which causes the meeting to be updated and the applicaton to come back to the state *ListMeetings*. An event of type *create* causes a transition to the state *CreateMeeting*, where a new meeting is created from data entered by the user. An event of type delete in the state *ListMeetings* triggers a transition that executes the action *deleteMeeting*, where the currently selected meeting is deleted from the database. Similarly, an event of type *cancel* causes the execution of *cancelMeeting*, which calls the method *cancel* on the selected meeting.

Model the above system by means of use-case diagrams, class diagrams and statechart diagrams, and possibly other type of UML diagrams that you think are appropriate.

QUESTION 2: Simple Calculator **[15 points]**

Draw a statechart diagram specifying the behaviour of the following simple calculator.

The interface of the simple calculator is composed of:

- 10 buttons with digits
- 4 buttons with the basic arithmetic operators (+, -, *, /)

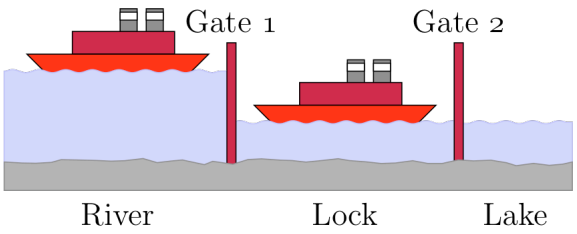
- The clear button "C" that resets the display
- The "=" button that evaluates the previously selected operation and displays the result
- Two buttons "On" and "Off".

The calculator is constrained as follows:

1) The calculator only supports operations with integers
2) Division by zero is reported with the message "ERROR"
3) The display has a capacity of 10 digits. The calculator must discard any digit entered after reaching the limit of 10 digits.
4) When the result of an operation requires more than 10 digits, the value is truncated to 10 digits by discarding the less significant digits (the digits on the right).

QUESTION 3: Modeling a ship lock **[15 points]**

The lock of Petriburg enables ships to go from a river with a high water level to a lake with low water level (ships use a different lock to go in the opposite direction). The lock has two gates, which both have two states: open and closed. The water level in the lock can be either high or low. On the river there is a place for one ship to wait until it can cross the lock, and in the lock itself there is room for at most one ship. The situation is sketched in the following figure.



The following must always hold:

- Gate 1 may only be opened, when the water level in the lock is high and gate 2 may only be opened, when the water level is low.
- After a ship has passed, the lock must always be able to let another ship pass.
- A gate is always either opened or closed. The water level is always either high or low. At all times, either a ship is waiting or not, and either a ship is in the lock or not.

Acknowledgment. This exercise was authored by Prof. Barbara König (University of Duisburg-Essen).