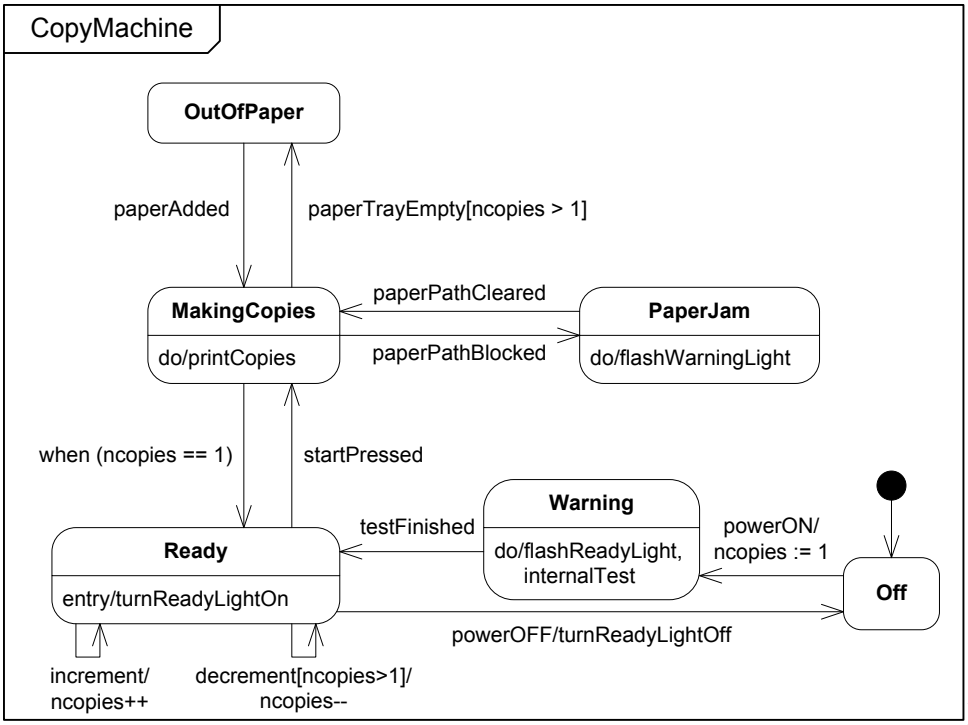
MTAT.03.083 – Systems Modelling

Final Exam – 8 December 2009

QUESTION 1: Graphical Document Editor **[10 points]**
Prepare a class diagram for a graphical document editor that supports grouping. Assume that a document consists of several sheets. Each sheet contains drawing objects, including test, geometrical objects, and groups. A group is simply a set of drawing objects, possibly including other groups. A group must contain at least two drawing objects. A drawing object can be a direct member of at most one group. Geometrical objects include circles, ellipses, rectangles, lines, and squares

QUESTION 2: Administration of Meetings **[15 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 3: Copy Machine **[20 points]**
Consider the following statechart diagram of a copying machine.



Initially The copy machine is initially off. When power is turned on, the machine reverts to a default state – one copy. While the machine is warming, it flashes the ready light. When the machine completes its internal testing, the ready light stops flashing and remains on. Then the machine is ready for copying.

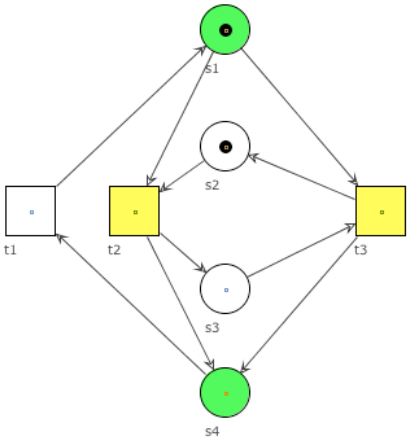
The operator may change any of the parameters when the machine is ready. The operator may increment or decrement the number of copies. Once the parameters have been set, the operator pushes the start button to start making copies. Ordinarily, copying proceeds until all copies are made. Occasionally the machine may jam or run out of paper. When the machine jams, the operator may open the lid and clear the paper path. When the operator closes the lid, the machine continues making copies. Adding paper allows the machine to continue copying after running out of paper.

Please complete the following two modifications of this state machine independently. In other words, starting from the statechart shown above, draw a statechart for sub-question (a). Then, starting again from the statechart shown above, draw another statechart for sub-question (b).

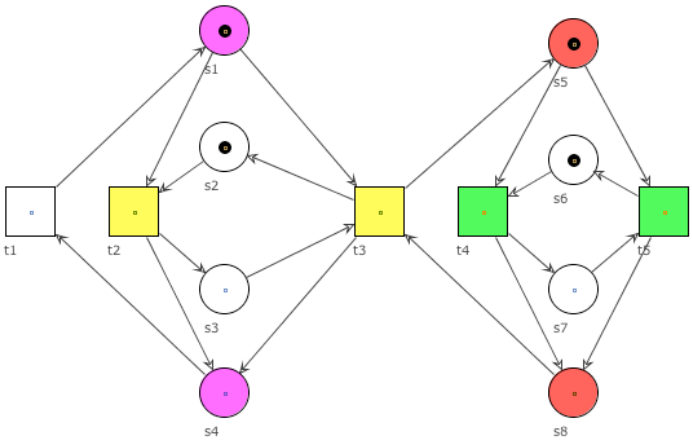
a. Refine the "MakingCopies" state to show a more detailed view of the process of making copies using a nested Statechart. At this more detailed level, specify when and how the variable "ncopies" (i.e. the number of copies) is updated during the copying process. In your answer, you only need to show the parts of the statechart that need to be modified.
b. Suppose that the copy machine jams and the operator turns it off. With the current specification, when the copy machine is turned on again, it moves to the Ready state, even if the paper path is still jammed. In addition, "ncopies" is set back to zero. Modify the diagram as follows: when the copy machine jams, the operator MUST first remove the jammed paper, then turn the machine off, and then turn it on again so that it can continue making the remaining copies. If the machine is turned off and on without first removing the offending paper, the machine stays jammed. And if the paper path is cleared while the machine is off, the machine will not be able to detect that the paper path has been cleared.

QUESTION 4: Petri Net Properties **[5 points]**
Consider the following three Petri nets.

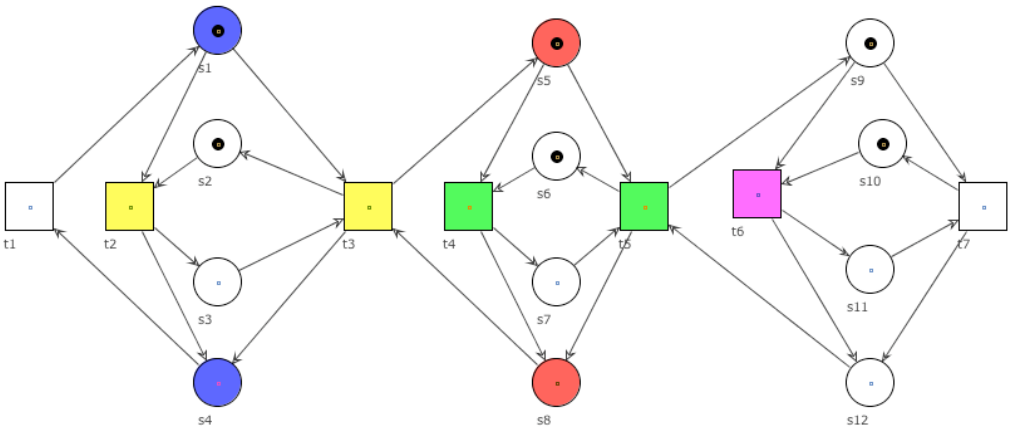
1



2



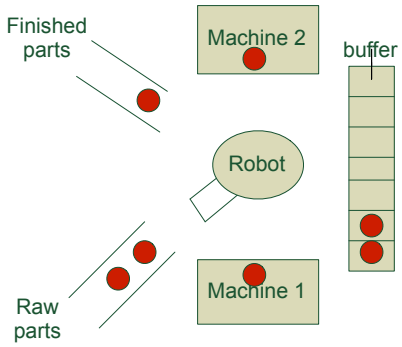
3



The three nets shown here are the first three elements of an infinite family of nets. We can create the n^{th} member of the family by taking the $(n-1)^{th}$ member and replicating the repeating pattern to the right. Consider the $\mathbf{n}^{th}$ Petri net in this family. Suppose that we fire **m** transitions on this net starting with the initial marking. Let us call **M** the marking resulting from this sequence of transition firings. Write a formula that computes the number of tokens in marking **M**. Justify your answer. Based on this formula, can you prove that the system is bounded?

QUESTION 5: Factory Line **[10 points]**

We consider a segment of a factory with two conveyor belts, two machines, one robot and one buffer. Raw parts arrive through a first conveyor belt, called the *raw line*. The robot moves each part from the *raw line* into machine M1, then into the buffer, then into machine M2 and finally to the second conveyor belt (called the *finished line*). M1 can hold at most one part at a time, and the same applies for M2. The robot can only move one part a time. The buffer can hold at most 7 parts. The conveyor belts can hold any number of parts. The following figure shows the case where the raw line holds 2 parts, M1 and M2 hold one part each, the buffer holds two parts, and the finished line holds one part.



From time to time, it happens that one of the machines breaks down. In this case the machine moves into a "broken" state and an alarm is triggered so that an operator fixes the problem. If the machine breaks down while it holds a part, the operator discards this part and restarts the machine without any parts inside. The robot must not touch the machine while it is broken. However, while one of the machines is broken, the other one may continue working.

a) Capture this system as a Petri net or as a Coloured Petri net.
b) Capture the same system as a Petri net, but this time assuming that M1 has a capacity of two parts (M2 still has a capacity of one part only). You only need to show how did you modify the places and transitions related to machine M1 (no need to redraw the rest of the Petri net). You can use either plain Petri net or Coloured Petri net notation.