UNIVERSITY OF TARTU

MTAT.03.083 – Systems Modelling

Regular Exam – 5 January 2015

Notes:

- The exam is open-book and open-laptop. Web browsing is allowed.
- You are not allowed to communicate with anyone during the exam in any way (except with the lecturer).
- You may submit your exam on paper or electronically. In the latter case, include all files of your submission in a zip file and submit it using the "Submit" button in the course Web page. Allowed file formats are: PDF, PNG and mdzip (MagicDraw).
- If you find that there is not enough information in the text below and you need to make additional assumptions, please write down your assumptions.

Part 1. Generating traces through simulation of BPMN models

A BPMN process model is a graph consisting of three types of nodes: *activities* (represented as rectangles), *events* (represented as circles), and *gateways* (represented as diamonds). An activity denotes a unit of work that needs to be performed. An event denotes something that happens instantaneously. Events and activities are always labelled with a name. Gateways serve to route the flow of control along the branches of the process model. We assume in this exercise that a BPMN model has one start event triggering a process execution, one end event to finish it, and no other events.

Nodes are connected by means of directed edges called *sequence flows*. A sequence flow basically says that the flow of control can pass from the source node to the target node. There can be at most one sequence flow connecting two nodes. In addition, only split gateways can have multiple outgoing sequence flows and only join gateways can have multiple incoming sequence flows. An illustrative example of a lending process is shown in Figure 1.

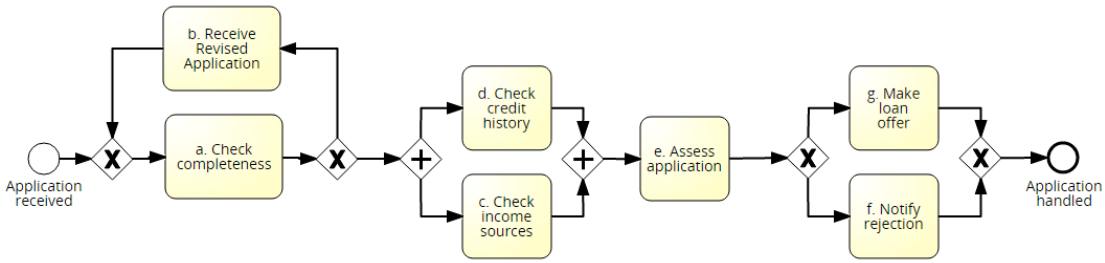

Figure 1

In this exercise we consider two types of gateways in BPMN: XOR gateways (represented by an 'X') and AND gateways (represented by a '+'). A gateway is said to be a *split gateway* if it has multiple outgoing flows. Conversely, a gateway is a *join*

gateway if it has multiple incoming flows. We will assume that a gateway is either a split gateway or a join gateway but never both. If we put together the distinction between the two types of gateways (XOR, AND) and the distinction between split and join gateways, we obtain four types of gateways: XOR-split gateway, XOR-join gateway, AND-split gateway, AND-join gateway.

An XOR-split is a decision point where the flow of control is passed to exactly one of the outgoing flows of the XOR-split. For example, Figure 1 has two XOR-split gateways: one after activity *a* ("Check completeness") and another after activity *e* ("Assess application"). An XOR-join on the other hand merges two incoming branches into a single one. Figure 1 features an XOR-join just before activity *a* ("Check completeness") and another one before the end event.

An AND-split forks out one thread of execution into two or more parallel threads. For example, Figure 1 has an AND-split that starts two threads in parallel corresponding to activities *c* ("Check income sources") and *d* ("Check credit history"). These two threads converge in an AND-join just before task *e* ("Assess application"). The AND-join is a synchronization point – it waits for both threads to complete.

A trace is a sequence of activity names (e.g., [x, w, y, a, g]). A valid execution trace of a process model is a trace generated by consecutively firing nodes in the process model according to the firing rules described below, starting from the sequence flow of the start event and ending in the sequence flow of the end event. Examples of valid execution traces of the process model shown in Figure 1 are:

- [a, d, c, e, g]
- [a, c, d, e, f]
- [a, b, a, c, d, e, g]
- [a, b, a, b, a, d, c, e, f]

On the other hand, the following are not valid traces of the process model in Figure 1.

- [a, d, e, g] ("c" is missing)
- [a, d, c, e, g, f] ("g" and "f" cannot occur in the same execution)

The simulation of a BPMN model is carried out through the so-called "token game", which defines when is a given node *enabled* (i.e., ready to fire) and what happens when a node *fires*. Each sequence flow of a BPMN model may hold zero or one token. When the simulation starts, the start event produces one token in the sequence flow coming out of it. An activity or a split gateway is enabled if the incoming sequence flow contains a token. An XOR-join gateway is enabled if one of its incoming sequence flows contains a token. An AND-join gateway is enabled if all its incoming sequence flows contain a token.

An enabled node can fire anytime. When an activity fires it consumes one token from the incoming sequence flow and produces one token in the outgoing sequence flow. When an XOR-split fires it consumes one token from the incoming sequence flow and produces one token in one of its outgoing sequence flows, randomly chosen. When an XOR-join fires it consumes one token from the incoming sequence flow that has a token, and produces one token in its outgoing sequence flow. When an AND-split fires it consumes one token from its incoming sequence flow and produces one token in each of its outgoing sequence flows. When an AND-join fires it consumes one

token from each of its incoming sequence flows and produces one token in its outgoing sequence flow. The simulation finishes when there is one token in the incoming sequence flow of the end event.

During a simulation, when an activity fires, the name of this activity is appended to the output trace. When a gateway fires, it does not produce anything into the output trace.

Task 1. [12 points]

Design a domain model (UML class diagram) of BPMN models and their simulation as defined above.

Task 2. [10 points]

Write a sequence diagram that describes how the classes in the domain model (and other classes if needed) interact in order to fire each type of node.

Task 3. [8 points]

Write a sequence diagram that describes how the classes in the domain model (and other classes if needed) interact in order to generate a valid trace.

Task 4. [5 points]

Design an application model for the above scenario. The application model should include methods required to generate a valid trace via simulation of a BPMN model.

Part 2. State chart

Consider the statechart diagram of a photocopying machine given in Figure 2.

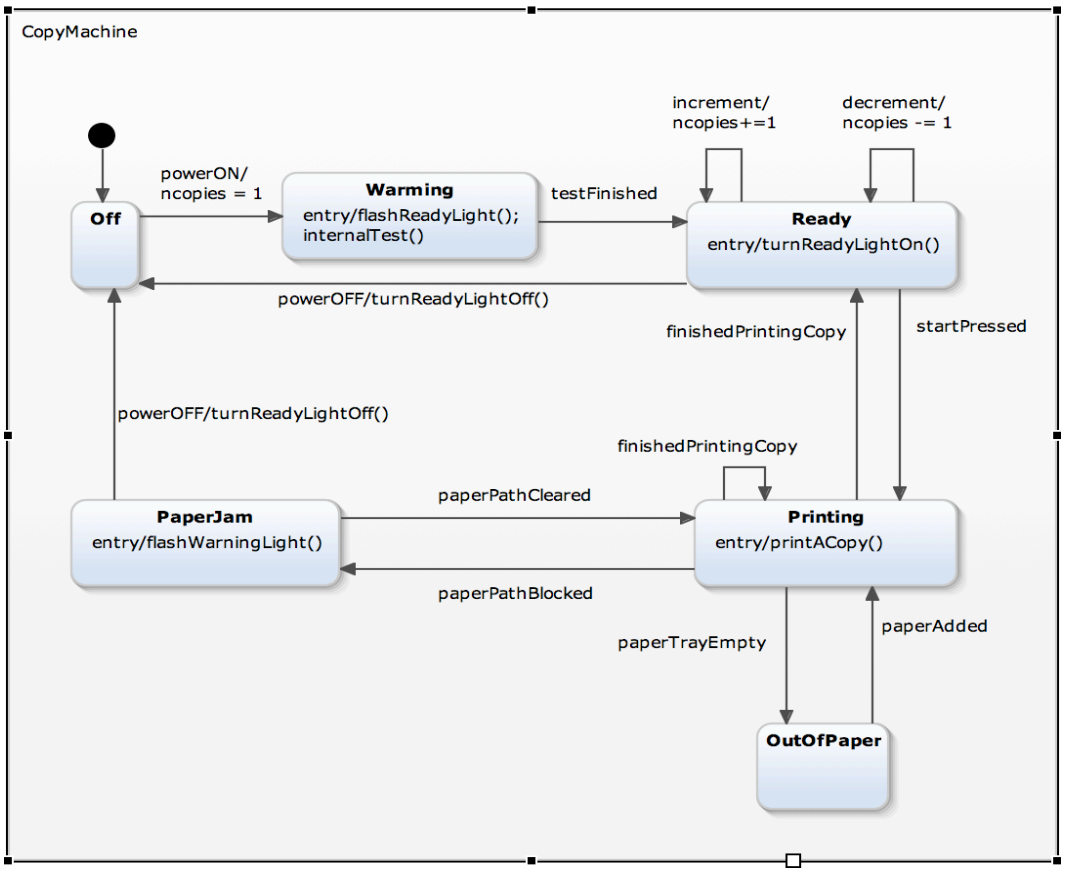

Figure 2

Initially, the photocopier is off. When the power is turned on, the photocopier goes to a warming state where an internal test is performed. While the photocopier is warming, it flashes the "ready" light. When the photocopier completes its internal testing, the ready light stops flashing and remains on. Then it is ready for copying.

Once the photocopier is ready, the operator may increment or decrement the number of copies to be made, and eventually pushes the start button. While photocopying, the machine may jam or run out of paper. When the machine jams, the operator must clear the paper path by opening the lid, removing the offending paper and closing back the lid. Similarly, when the machine runs out of paper, the photocopier stops copying until the operator adds more paper.

Task 5. [15 points]

Modify this statechart as explained below.

a. The current statechart does not count the number of copies that are made. Modify the statechart so that it makes the number of copies "ncopies" specified by the operator. To this end, introduce a counter "copiesMade", which keeps track of how many copies have been made so far. Add guards and actions as required to achieve the desired behaviour. **[5 points]**
b. Suppose that the photocopier jams and the operator turns it off. According the current statechart, when the photocopier is turned on again, it moves to the warming and then to the "ready" state, even if the paper path is still jammed. In addition, "ncopies" is set back to one. Modify the statechart as follows: when the photocopier jams, the operator MUST first remove the jammed paper, then turn the photocopier off, and then turn it on again so that it can continue making the remaining copies. If the photocopier is turned off and on without first removing the offending paper, it 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. **[10 points]**

NOTE. The given statechart does not specify what happens when the machine is powered off in the "Warming", "Printing" and "OutOfPaper" states. You are not asked to modify the statechart to specify what happens when powering off the photocopier in these three states. The behaviour of powering-off in these three states is outside the scope of this exercise.

WHAT TO SUBMIT: If submitting electronically, the statechart must be submitted as a PDF of a PNG file (no Yakindu or MagicDraw files).