BCS2213 – Formal methods

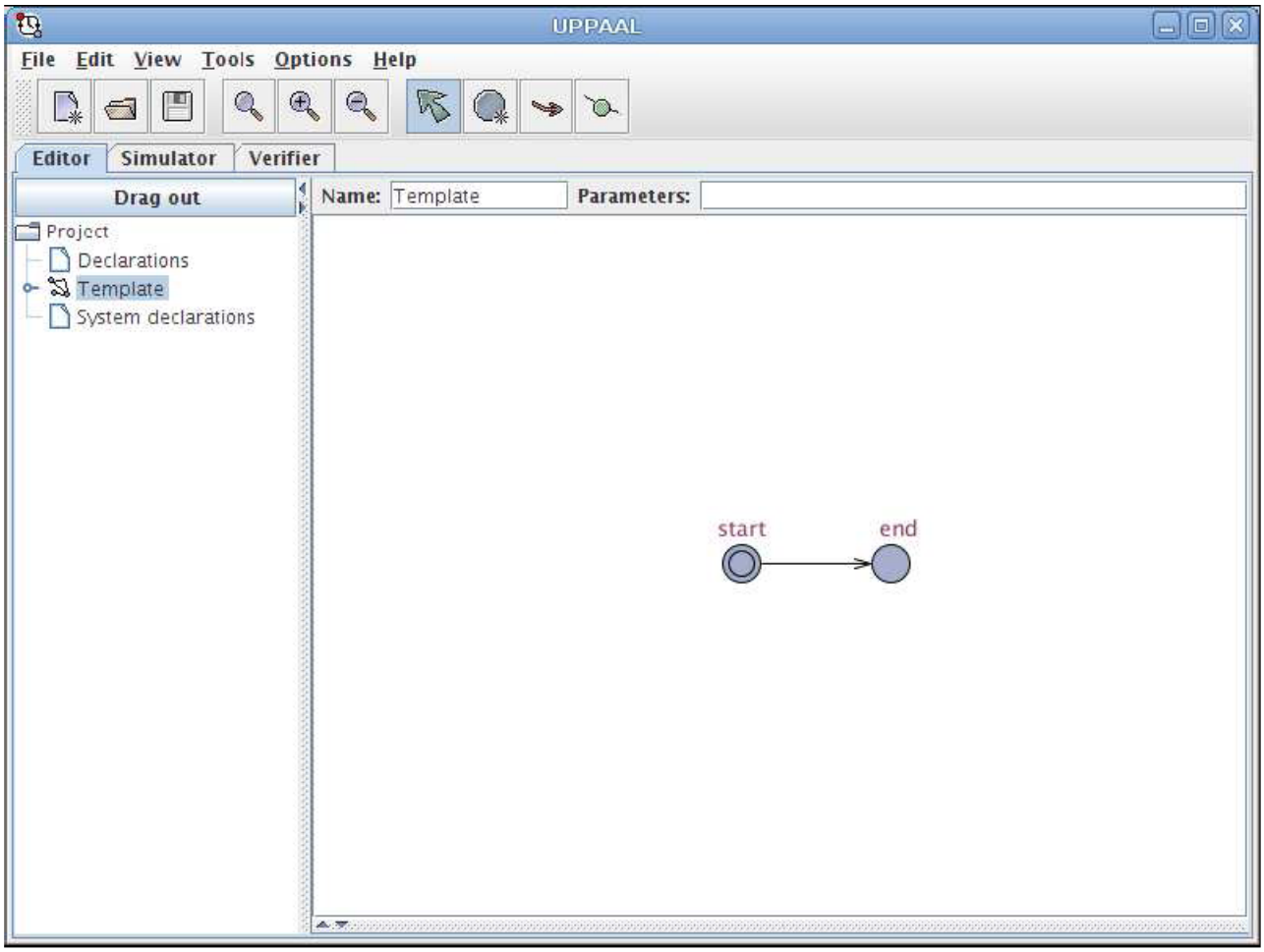
Teaching assignment 5. Learning UPPAAL. Modelling systems using timed automata.

1. Run UPPAAL Toolbox.

UPPAAL is a Toolbox for *validation* (via graphical simulation) and *verification* (via automatic model-checking) of real-time systems. It consists of two main parts: a graphical user interface and a model-checker engine.

The UPPAAL main window has two main parts: the menu and the tabs - the **editor**, the **simulator** and the **verifier**.

2. Editor mode. When you start UPPAAL, there is one *location* pre-created in the drawing area. It is the *initial* location of the automaton (template), so it has an additional circle inside. To add a second location, click on the "Add Location" button in the tool bar (tool tips will help you) and then click in the drawing area, a bit next to the initial location. To exit from the location creation mode click the "Selection Tool" button. Double click on location nodes to give them the names **start** and **end**. Then choose the Add Edge button, click on the **start** location and on the **end** location (see figure below).

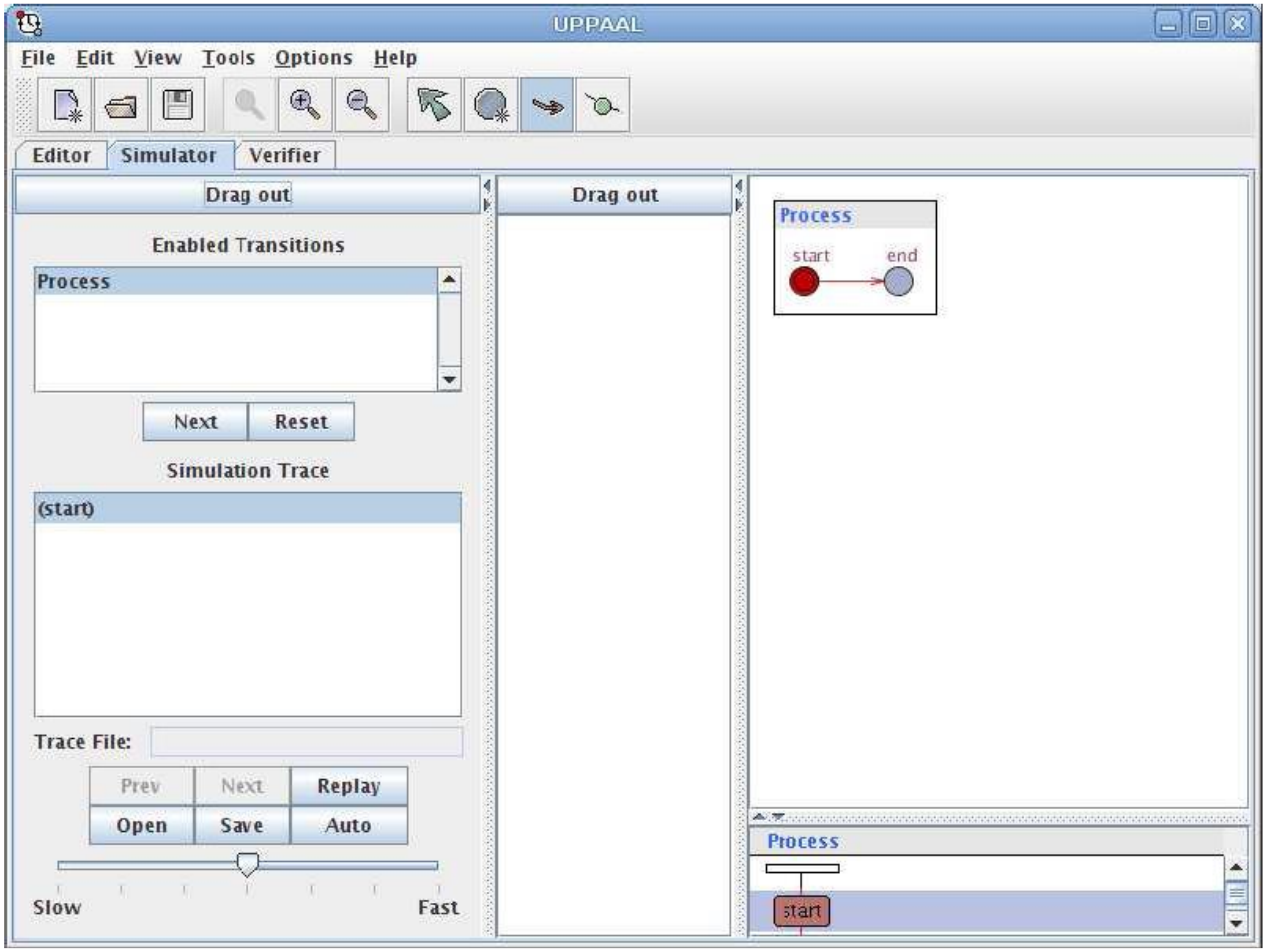


Note, by clicking on the corresponding items in left pane of UPPAAL you may edit project and systems declarations.

3. Simulator mode.

Now, click on the Simulator tab to start the simulator, click on the *yes* button that will pop up and you are ready to run your first system.
Figure below shows the simulator view. On the left you will find the control part where you can choose the **enabled transitions** (upper part) and work on an existing **simulation trace** (lower part). In the middle are the **drag out** widget for the variables (we have no for the moment) and on the right you see the simulating system itself. Below the system, you will see simulation trace of **process** shown as *message sequence diagram*.

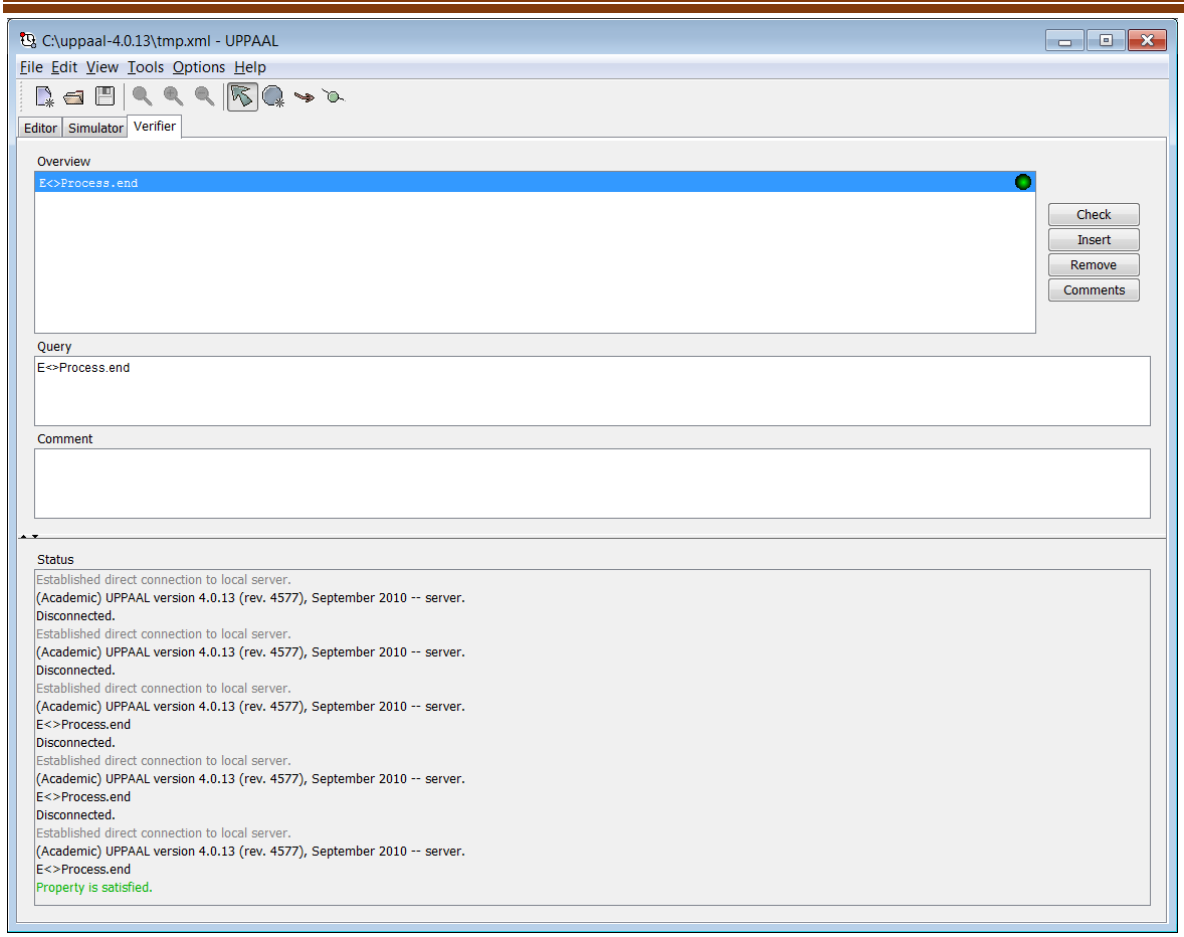
To simulate our trivial system pick one of the **enabled transitions** in the list in the upper left part of the screen (there is only one transition in our example). Click Next. The process view to the right will change (the red dot indicating the current location will move) and the simulation trace will grow. You will note that more transitions are not possible, i.e. the system is *deadlocked*.



4. Verification mode.

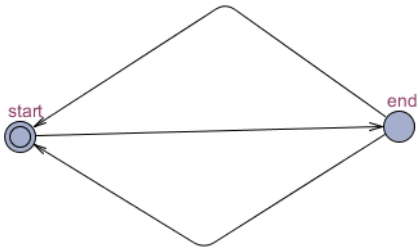
Click on the **Verifier** tab (see figure below). The upper section allows you to specify *queries* to the system. The lower part **Comments** the communication with the model-checking engine.

Enter the text E<>Process.end in the *Query* field below the Overview. This is the UPPAAL notation for the temporal logic formula ∃◊Process.end and should be understood as "it is possible to reach the location **end** in the automaton **Process**". Click Check to let the engine verify this. The green bullet in the overview will turn green indicating that the property indeed is satisfied.



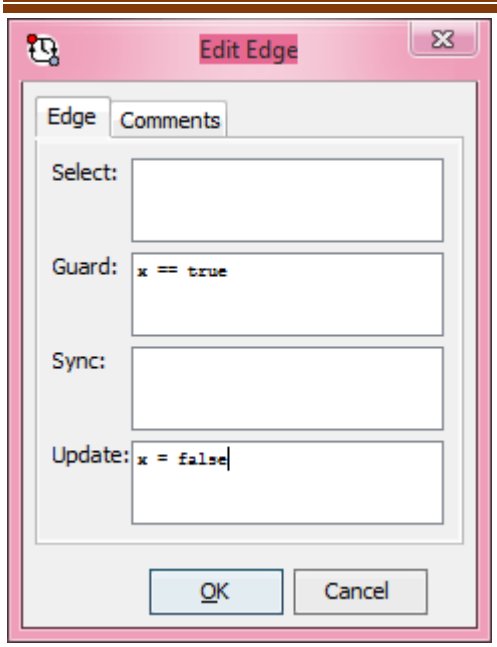
5. Modelling processes that are more complex.

Modify your model in order it has two possible transitions from the **end** to the **start** location (see figure below).



In the Project Declarations define Boolean variable x (bool x;), and allow the transition from start to end node if x is false, and from end to start node – if x is true (so you need define *guards* for these transitions, correspondingly x==false, x==true). *Updates* for the state transitions will be correspondent assignments (x=true, x=false).

Note, to access *guard* and *update* sections you need double click on the corresponding edge (see figure below).



Track your model in simulation mode. Note, that initial (automatically assigned at initialization) value of x is 0 (false), so initial transition from start to end location is allowed.

Make one screenshot from Editor and one from Simulator window of this model and put into Word file with your explanations.

Save the file with name lab_5_<your_ID>.doc(x)

6. Develop the model of Intelligent Light Control system.

Requirements:

- If a user presses the light control, then it lights;
- If a user *quickly* presses the light control, then the light should get brighter;
- if the user *slowly* presses the light control, the light should turn off.

Note, the steps for the development of this model were considered in details in our Lecture №9. Introduction to UPPAAL.

6.1. Track your model in simulation mode.

6.2. Make screenshots of the model of Intelligent Light Control system (Editor and Simulator pages of UPPAAL to be presented) and update your Word file, adding possible comments.
Note, a lab sheet reusing screenshots captured from the lecture will not be accepted.

6.3. Check the property of the Light Control system**: A[](x>=0)**
Explain this property by making comment in a Word file.

6.4. Define own temporal property of the Light Control system and check it with UPAALL.
Explain this property by making comment in a Word file.

7. Upload lab_5_<your_ID>.doc(x) into Kalam.

This assignment will be evaluated in maximum 2.5% of your total marks.