Instructions to use PIPE+

PIPE+ is implemented under the environment of Eclipse. User can import it into eclipse to view source code as well as run the program. The user can choose RunGUI to run the tool.

Since PIPE+ is a high level Petri net modeling tool including an editor and a simulator. We are going to present how to use the tool through an example.

Modeling:

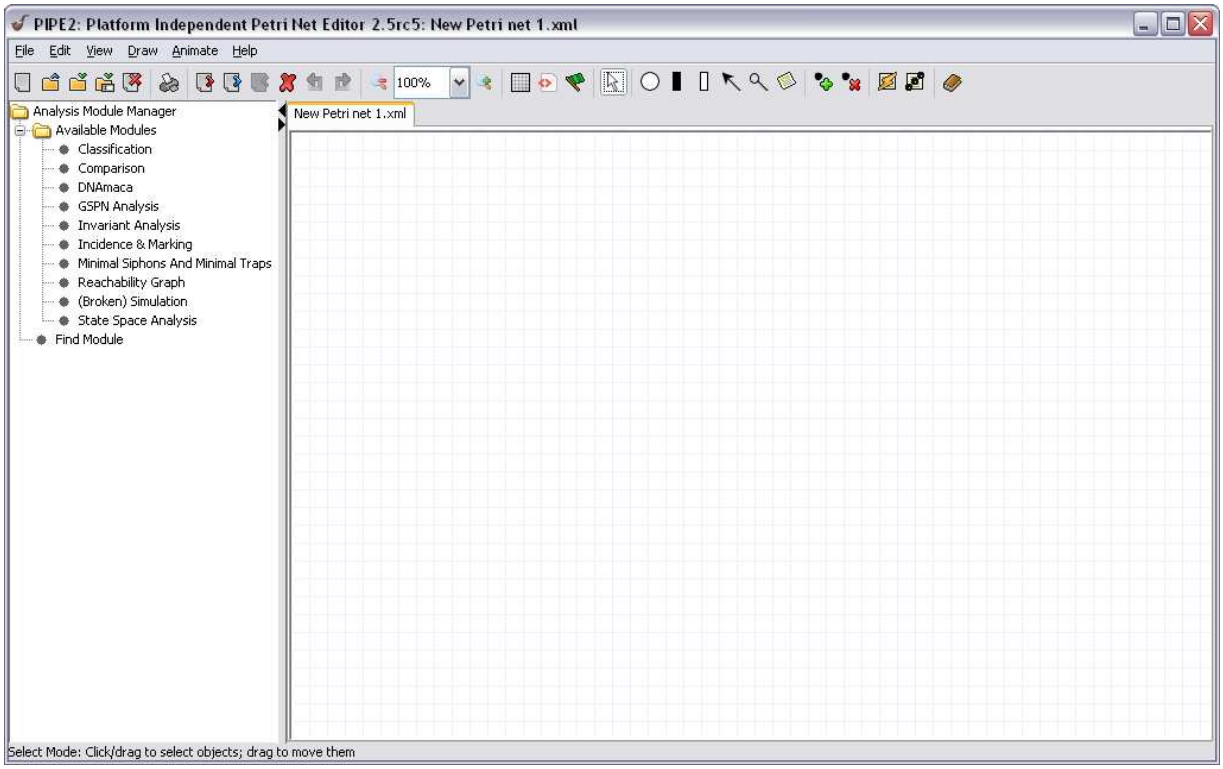


Figure 1 Editor's Screenshot

1. Brief view of the PIPE+'s interface:
    a. When you run the tool, Fig.1 is the first interface you will see. The upper part of the screen frame is menus and button thumbnails. They perform the same functionalities on modeling process, so you can view them as synonymy;

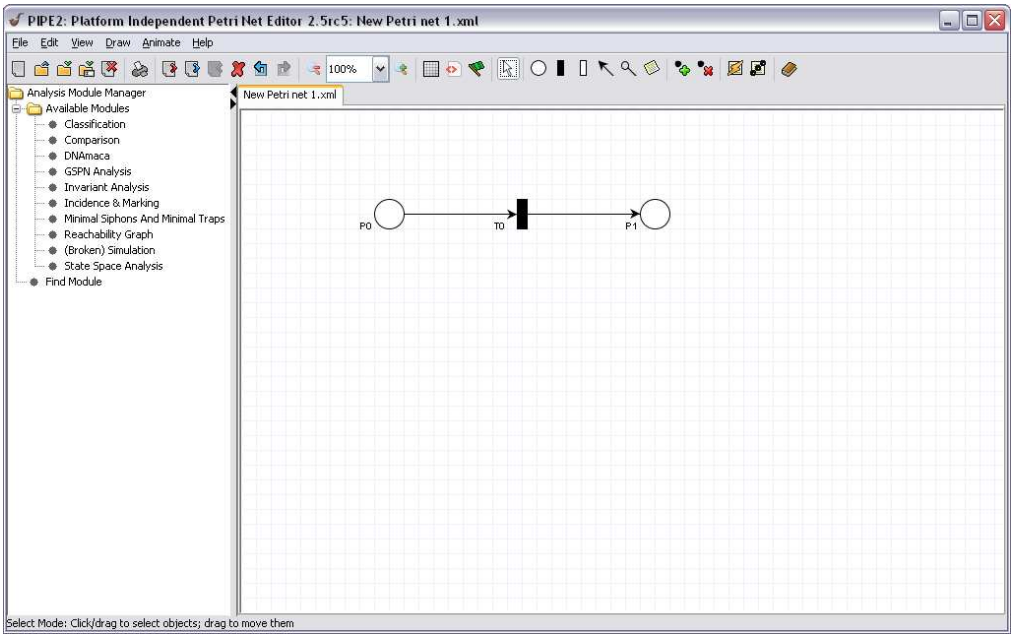
b. Note: since PIPE+ is built based on PIPE (a low level Petri net tool), several buttons of low level Petri nets' elements are still retained and may not be used when modeling high level Petri net;

c. The buttons with graphic thumbnail are listed under the menu. You can move your mouse pointer onto the thumbnail and an instruction is appeared informs you what is the use of the button;

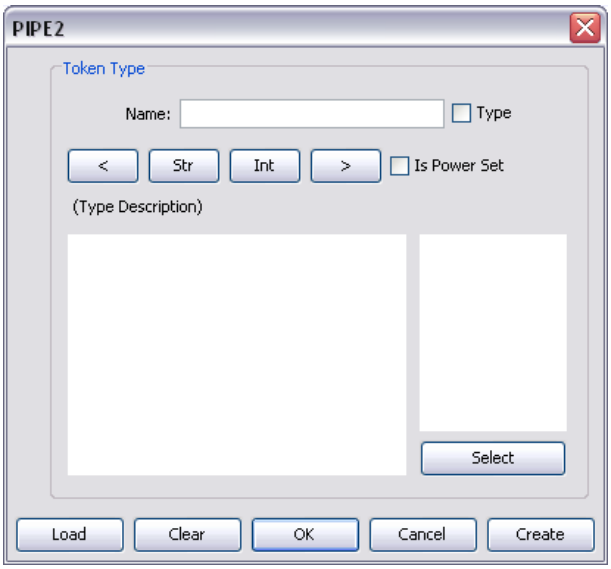
2. Build a net

a. Since at the time the interface is displayed, a new blank file is automatically built. User can also build another new file by clicking the button "new" or open an existing file (for example the xml file called "MondexConcreteModel") by clicking "open" button;

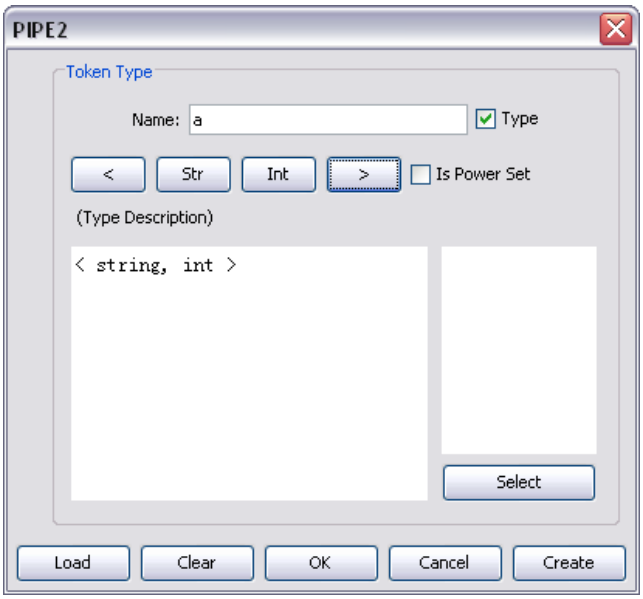
b. In this example, we draw two places and one transition, and we draw arc connecting places to transition.



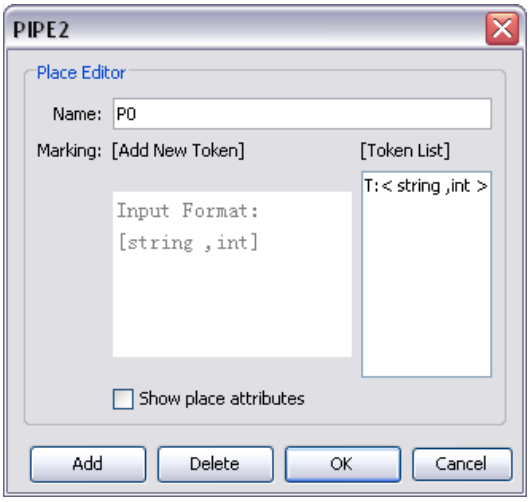
c. Then we define each places by right click the place circle and choose "define type", take the place P0 as an example, then we see a dialog below:



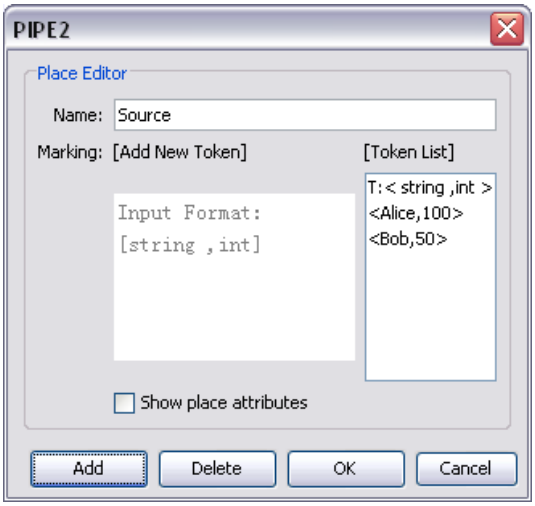
The figure above is a dialog for place type definition. We input name “a”, click the “type” button indicates the place is defined by now. Then, we define the place to be <string, int> by clicking the button at the second line. We leave the “is Power Set” blank because we do not want this place to be a power set. Otherwise, click the “is Power Set” button indicates it is a powerset. The defined place type is now like the figure below:



After place type definition, we can add tokens into the defined place by right click place circle and choose “edit place”, and we see another dialog like the figure below:

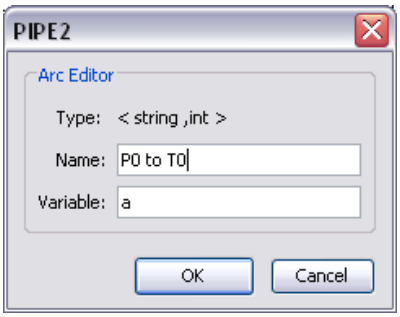


From the figure above, we can modify the place name by changing the text box of name, for example we changed it to “Source”. Then we see the text box of add new token, we see the type we just defined [string, int], so we type “[Alice, 100]” then we click “add” button, the button added will be displayed at the token list text box. We add another token “[Bob, 50]” in this place. After add tokens, the dialog is like the figure below. Then we click “OK” to exit this dialog.

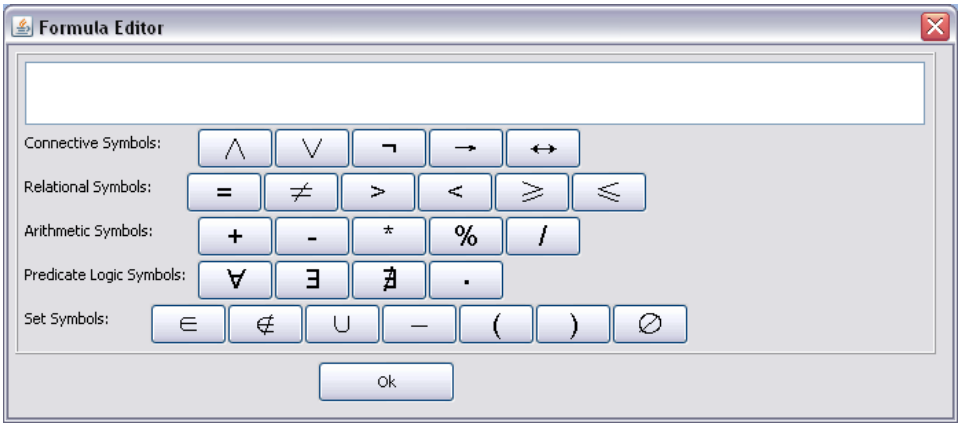


Note: we should define all the places’ type before simulation, otherwise will cause error.

d. Then we define arc variable. We choose the arc and right click to pop up a menu, then choose edit variable. We input “a” into the text box of variable. Note, by convention, we input low level char indicating the connected place is not power set while high level char indicating it is a power set.



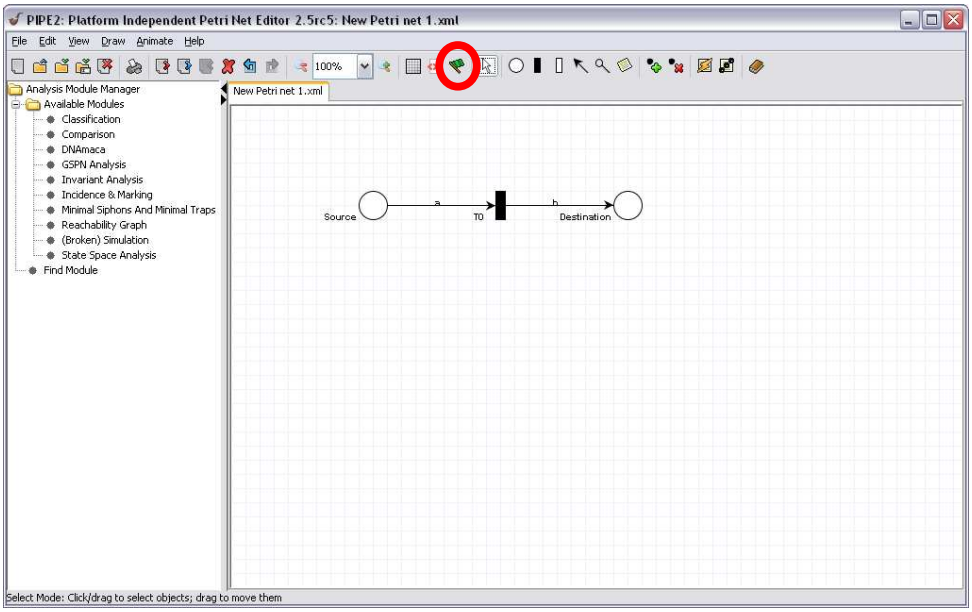
e. After that, we define transition condition by right click the transition to pop up a menu and choose “add/eidt formula”. Then we see an editing panel with a text area for us to input first order logic.



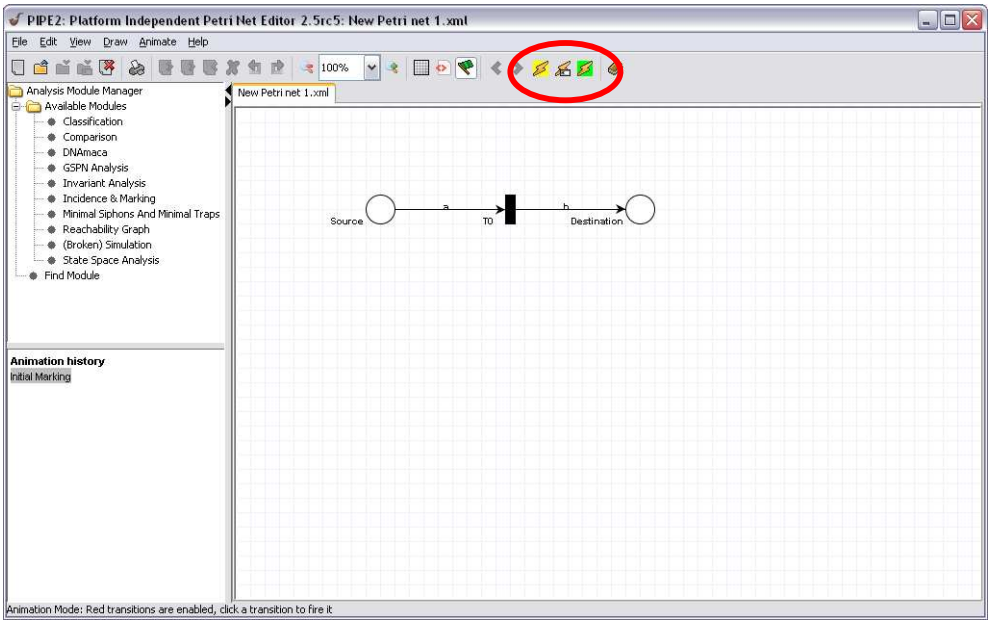
Since the logic formula has a lot of symbols that cannot be input by keyboard directly, we can make use of the symbol’s panel. We defined the formula as below, means if the second element of a less than 100, then we send the token from a to b’s place. Note: In this case, since b=a, place connects to b arc should has the same place type with place connects to a arc.

3. Simulation:

   a. At the time when we finished build a simple net model, the we see the result below:



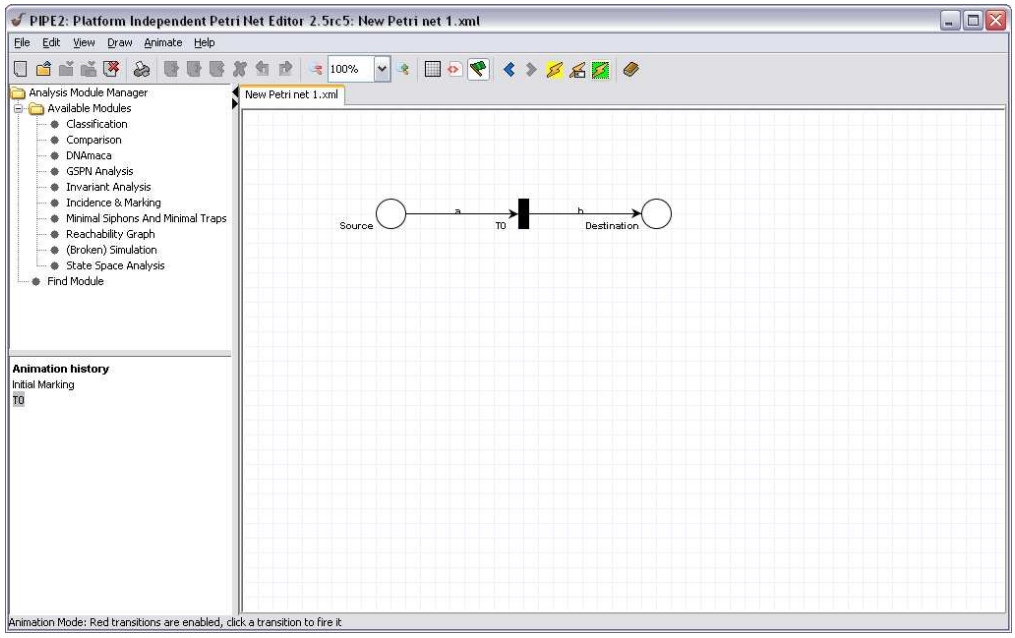
   We click the “green flag” button (highlighted in a red circle in the figure above) to switch to animation mode.



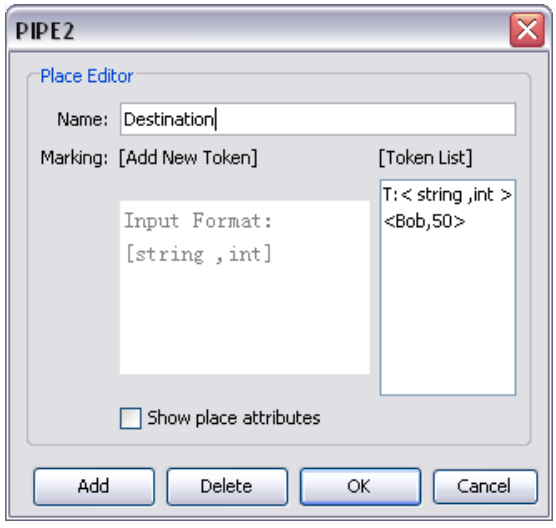
   b. After switched to animation mode, the menu above is changed by substituting buttons control the simulation (highlighted in red circle). From the figure above, to randomly fire a transition, we click the button with green background to fire a high level Petri net.

Note the button with yellow background is useless here because we modeled a high level Petri net on the canvas.

c. Since the example net only has one transition and the source place only have two tokens with only of them satisfy the transition condition, which can be fired. The simulation history is only fired one transition T0, see the figure below.



We can view the token flow by switch back to editing mode by click the “green flag” button. Then right click Destination place, and choose edit place to view place token list, see the figure below.



From the figure above, which is the tokens in the place “Destination”, we can see the token [Bob, 50] previous from the place “Source” now appeared in the “Destination” token list. The simulation result is the same we expected.