1. Features that have been implemented
A basic Petri net interface has been implemented, along with a simple evaluation tool. The tool includes a debug mode and a Simulation mode. Debug mode allows the user to walk step by step through the execution of a Petri net. The user is in control of which active transition to fire. The user may also inject tokens into the net at any time. The screenshot below illustrates debug mode.
In simulation mode, the user specifies the initial marking of a Petri net, and then the Petri net executes automatically for the user to observe. In addition, support for a token generator, which can supply a particular place with tokens during the execution of a Petri net, has been implemented.

```
0: Debug Mode
1: Simulation Mode
0
Places with tokens:
Active transitions:
Provide an initial marking
Enter the places in which you would like to insert a token.
Gesture received
No prior gesture received

Places with tokens:
'No prior gesture received' has 1 token(s)
'Gesture received' has 1 token(s)
Active transitions:
Send help message
Select an active transition to fire.
Active transitions:
0: Send help message
other: Choose non deterministically
0

Places with tokens:
'Manager received help message' has 1 token(s)
'No prior gesture received' has 1 token(s)
Active transitions:
Call patient
Enter the places in which you would like to insert a token.
```

2. Features yet to be implemented
Additional debugging functionality, such as breakpoint features, will be implemented. In a Petri net, breakpoints may be defined by markings, which represent the state of the Petri net.
In simulation mode, statistics such as marking frequency, range of values of tokens, and the firing frequency of a particular transition, will be recorded. In addition, colored Petri nets will be supported. Tokens will have values, places will have data types, and transitions will have conditions. Finally, an easy to use GUI will be implemented. Users will define the basic structure of the Petri net using the PIPE editor, which does not support all of the features of colored Petri nets, and then import that design into this tool. The user will then use this tool to augment that design with the features of colored Petri nets. An extension of the PNML XML grammar will be used to store colored Petri nets.