Chapter 11. Finite State Machine Domain

Authors: *Bilung Lee*

Other Contributors: *Christopher Hylands*
Mary Stewart

11.1 Introduction

The finite state machine (FSM) has been one of the most popular models for describing control-oriented systems, e.g., real-time process controllers. A directed node-and-arc graph, called a state transition diagram (STD), can be used to describe an FSM. An STD represents a system in the form of states (nodes) and transitions (arcs) between states.

11.2 Graphical User Interface

The original visual interface to Ptolemy, called Vem, is not suitable for drawing the STD for an FSM. A new visual editor based on Tycho, a hierarchical syntax manager and part of the Ptolemy project, is being developed based on drawing mechanisms created by Wan-teh Chang.

11.2.1 Edit a new STD file

A new STD file can be created from a Tycho editor that is running inside Ptolemy or from a standalone Tycho process. To start Tycho from within Ptolemy, type a `y`. To start a standalone Tycho process, run `tycho` from the shell prompt. Once Tycho is started, either open a file with the file extension `.std`, or select the Tycho `Window` menu and choose `State transition diagram editor`. A message window will pop up to ask for machine type. Currently there is one type of FSM supported, called mixed `Mealy/Moore machine`.

11.2.2 Edit the Input/Output and Internal Events Names

If there are any input/output and internal events for the FSM, their names must be specified as follows: Select the `Special` menu and choose `I/O Port Names...`, then enter the names for the input/output. Each name should be separated by at least one space.

11.2.3 Draw/Edit a State

To draw a state, either select the `Edit` menu and choose `New Node`, or use the keyboard accelerator `N` key. A crossbar cursor will appear in the window. Press and hold (don't release) the left mouse button and move the mouse to get a different shape of node. Release the button to finish the drawing.

To edit a state, first select the state by pressing the left mouse button on your selection. Then either select the `Edit` menu and choose `Edit Item`, or use the keyboard shortcut and type an `e`.

11.2.4 Draw/Edit a Transition

To draw a transition, either select the `Edit` menu and choose `New Arc`, or use the keyboard shortcut and type an `A`. A crossbar cursor will appear in the window. Press the left mouse button on the periphery of the starting state. Move the mouse and press the left mouse button to get a more delicate arc. To finish the drawing, move the mouse on the periphery of the ending state and press the left mouse button.

To edit a transition, select the transition by pressing the left mouse button. Then either select the `Edit` menu and choose `Edit Item`, or use the keyboard shortcut and type an `e`.

11.2.5 Delete a State/Transition

First select the state/transition by pressing the left mouse button on your selection. Then select the `Edit` menu and choose `Delete`.

11.2.6 Move/Reshape a State/Transition

To move a state, press and hold the left mouse button on the state and then move the mouse to the desired position. To reshape a state, first select the state, and then move the cursor to the up/down/left/right periphery of the state to get an up/down/left/right arrow-shape cursor. Press and hold the left mouse button and move the mouse to reshape it.

To move/reshape a transition, first select it, then some small rectangles will appear along the arc. Select and move the rectangles to move/reshape the arc.

11.2.7 Slave Processes of States

In an FSM, each state may be associated with a slave process. This slave process could be a subsystem of any other Ptolemy domain or be another FSM Galaxy. To specify the slave process of a state, when editing a state, give the full path name and file name of the subsystem. For example, a Vem facet could be specified for the Galaxy of any other Ptolemy domain, or an STD file could be specified for the FSM Galaxy.

11.3 Working within Ptolemy

In terms of implementation, a standalone FSM domain in Ptolemy is not very interesting. The reason for this is that most applications, in addition to control, contain many other features, e.g., signal processing. Moreover, there are other Ptolemy domains with which an FSM can interact. By mixing the FSM domain with other domains, we get a very powerful FSM model. Currently the FSM domain only works with the SDF and DE domains.

11.3.1 Make an Icon in Vem

Unlike other domains, an FSM galaxy is edited in a `.std` file using Tycho instead of Vem. However, to work with other domains, an icon in Vem is required to represent the corresponding FSM galaxy described in an STD file. To make an icon in Vem for the STD file, first start Tycho from within Ptolemy by pressing `y`. Open the file in Tycho and choose `Make Icon...`from the `Special` pull-down menu. This will load the FSM galaxy into the Ptolemy kernel and generate an icon with appropriate input/output ports in the specified palette in Vem.

The icon looks like a star (blue outline) but it is actually a galaxy. This may be confusing, but the idea is to avoid using an Octtools handle. There are two different ways to make an icon in Vem: `ptkSetMkSchemIcon` and `ptkSetMkStar`. The former needs an `OctFacetHandle` as one argument and is used for other "Vem" galaxies. However, since the FSM galaxy uses Tycho as the editor instead of Vem, there is no `OctFacetHandle`. Therefore the latter (`ptksetMkStar)`, which uses the star (or galaxy) name instead, is more appropriate.

11.3.2 Look Inside an FSM Galaxy

Similar to the galaxy icon in the other domain, when the `i` key is pressed on the icon to look inside the galaxy, the corresponding STD file which describes the FSM galaxy will be automatically invoked in Tycho. (Note: if the environment variable `PT_DISPLAY` is set to another editor instead of Tycho, it must be unset for the look-inside to work.)

11.3.3 Compile an FSM Galaxy

The FSM domain uses EditSTD to edit a Galaxy, but other domains use Vem as the editor. Therefore, when an FSM Galaxy is compiled, the EditSTD (Tycho) is invoked to compile the state transition graph into the Ptolemy kernel.

11.4 An overview of FSM demonstrations

There are currently three FSM demos illustrated in figure.

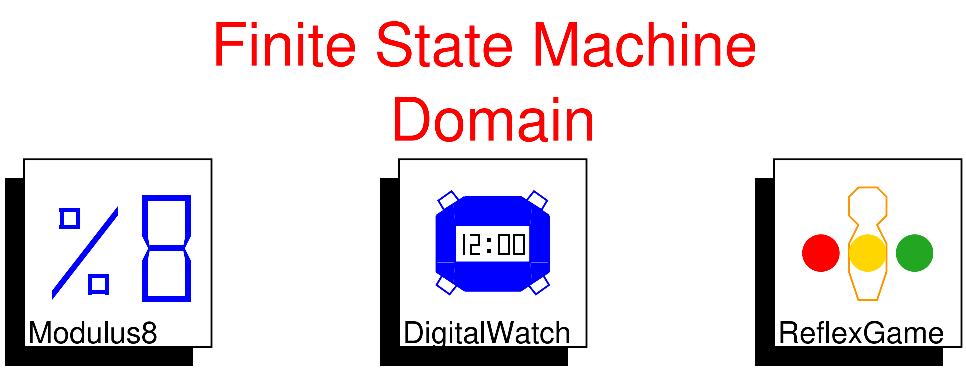

FIGURE 11-1: Finite State Machine demos.

`modulus8` A three-bit counter with initialization and interruption mechanisms. This demonstrates how the three-bit counter is built by three one-bit counters communicating in the SDF domain.

`reflexGame`	A simple game to test the reflexes of the player. The top-level DE domain models a clock, while the FSM domain models the various states in the reflex game. The SDF domain is used for numerical computations.
`digiWatch`	A digital watch example. This demonstrates that a system with sophisticated control can be achieved by hierarchically nesting the FSM domain.

11.5 Current Limitations

As of Ptolemy release 0.7, this domain is still experimental and is not yet fully developed. Under current implementation, the FSM domain can only be used as a Galaxy. Thus, the FSM domain has to work with other domains that support the inputs to the FSM Galaxy and display the outputs generated by the FSM Galaxy. However, it is possible to implement some blocks that provide the inputs and that display the outputs in the FSM domain.

An FSM system terminates when it reaches a final state, but this feature is not yet implemented. Further, since the FSM system must currently be embedded in other domains under current implementation, the termination of a system is controlled by the topmost domain.