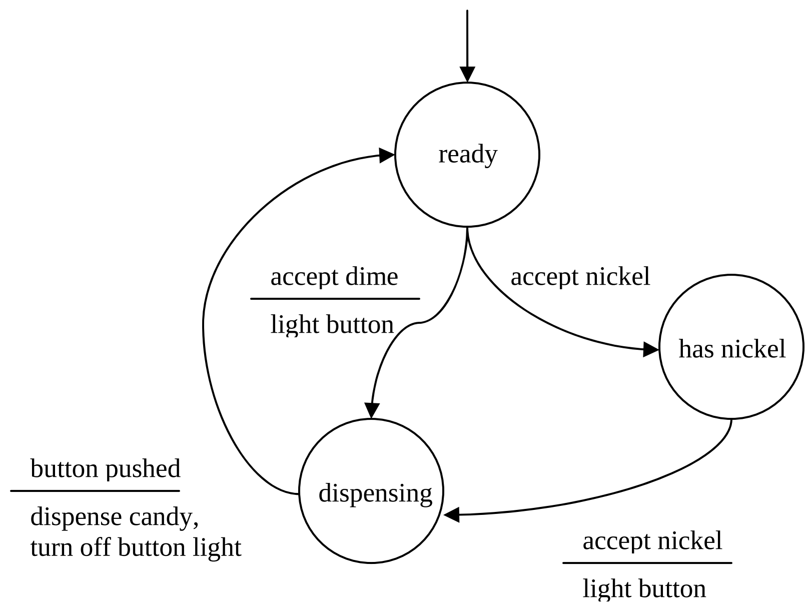
Protocols

Protocols are the rules governing the interaction between two (or perhaps more) parties. Societies – both human and animal – establish protocols of social behavior through cultural practices. These protocols may distinguish the relative social standing of the parties and define the accepted forms of exchange among them. Interactions among or with technical artifacts or systems may also be expressed as a protocol. For example, a communications protocol defines the rules which computing systems use to exchange data. A heavily used protocol on the internet is the TCP (Transmission Control Protocol).

Protocols related to technical artifacts and systems can be represented using finite state machines. The document that defines the TCP protocol, for example, uses a finite state machine to clearly express the behavior of the protocol. When used in this way, a state of a finite state machine represents the current stage of the protocol. The conditions associated with the transitions from this state give information about what is required for the protocol to enter its next stage.

A finite state machine model of a reactive system can also be viewed as defining the protocol for interacting with the system. The terms of the protocol are conveyed by the conditions on transitions that require the operation of an external entity. For example, in the simple candy dispenser presented earlier (and duplicated below) the “accept nickel” and “accept dime” conditions are expressed as conditions which, when satisfied, causes the transition to a new state.



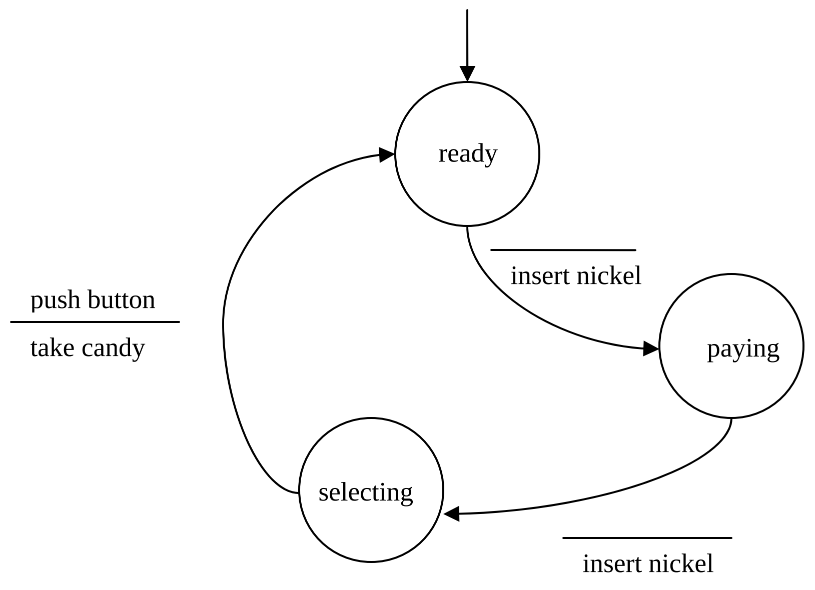
Candy dispenser finite state machine

Our understanding of the finite state machine for the candy dispenser was used to say that among the possible behaviors of the candy dispenser were:

accept nickel, accept nickel, light button, button pushed, turn off light, dispense candy
accept dime, light button, button pushed, turn off light, dispense candy

where the conditions which must be satisfied by an external agent are underlined. This distinction between actions that the artifact or system can take on its own and those conditions that are caused by external agents is important and realistic. For example, we would not expect a candy dispenser to insert coins into itself; it waits for a customer to insert the coins. Also, we would not expect the customer to switch on the light that signifies that sufficient payment has been received; the candy dispenser is built to do that on its own. The dispenser is also built to turn off the light and to dispense the candy. However, the customer is expected to take the candy (but whether the customer does so or not is not the concern of the candy dispenser).

By modeling the behavior of the candy dispenser's customer the "other half" of the protocol can be seen. The finite state machine for a simple customer is shown in the figure below. This simple customer always pays with two nickels.



Simple customer finite state machine

The labels on the transitions for the simple customer's finite state machine have been chosen to reflect their relationship to the corresponding conditions in the candy dispenser's finite state machine. For example, the *insert nickel* action of the simple customer satisfies the *accept nickel* condition in the candy dispenser. Similarly, the *push* *button* action of the simple customer satisfied the *button pushed* condition in the candy dispenser.

Notation: We can be explicit about the relationship between the actions and conditions of the interacting finite state machine by using some simple notation like

(Candy dispenser:accept nickel, Simple customer:insert nickel)

which is interpreted to mean that when the simple customer performs the insert nickel action the accept nickel condition in the candy dispenser is satisfied.

Notice that the candy dispenser has actions and behaviors that are not relevant to the simple customer:

- the simple customer does not exercise the behavior of the candy dispenser that allows payment by a single dime, and
- the simple customer does not pay any attention to the light that the candy dispenser turns on and off.

The customer may also perform actions that are not relevant to the vending machine (see the exercises below).

The important connection to see is how the two finite state machines interact – each can perform actions that enable the other to move to a new state. In this way the two finite state machines "co-evolve". This co-evolution is possible because they have a common protocol. The combined (interleaved) behavior of the candy vending machine and the simple customer can be written as:

insert nickel, accept nickel, insert nickel, accept nickel, light button, push button, button pushed, dispense candy, turn off button light, take candy

This behavior can, of course, be repeated indefinitely.

Exercises

1. Develop a variation of the simple customer where the customer waits for the button light to be turned on before the customer pushes the button.
2. Develop a finite state machine for a customer that pays only with a dime. Write the combined (interleaved) behavior resulting from this customer's interaction with the vending machine.
3. Develop a finite state machine for a customer that can pay with either dimes or nickels. Write the two combined (interleaved) behaviors resulting from this customer's interaction with the vending machine.
4. Develop the finite state machines for two customers who each insert a nickel. One of the customers pushes the button and the other takes the candy. The customer who takes the candy divides it in half and gives one half to the other customer. (Think

about this question: does the vending machine know that there are two customers? Does it matter?)

5. Develop the finite state machine for a customer that inserts a nickel and then inserts a dime. What is the combined (interleaved) behavior resulting from this customer's interaction with the vending machine? What do you think this means in terms of their protocol?