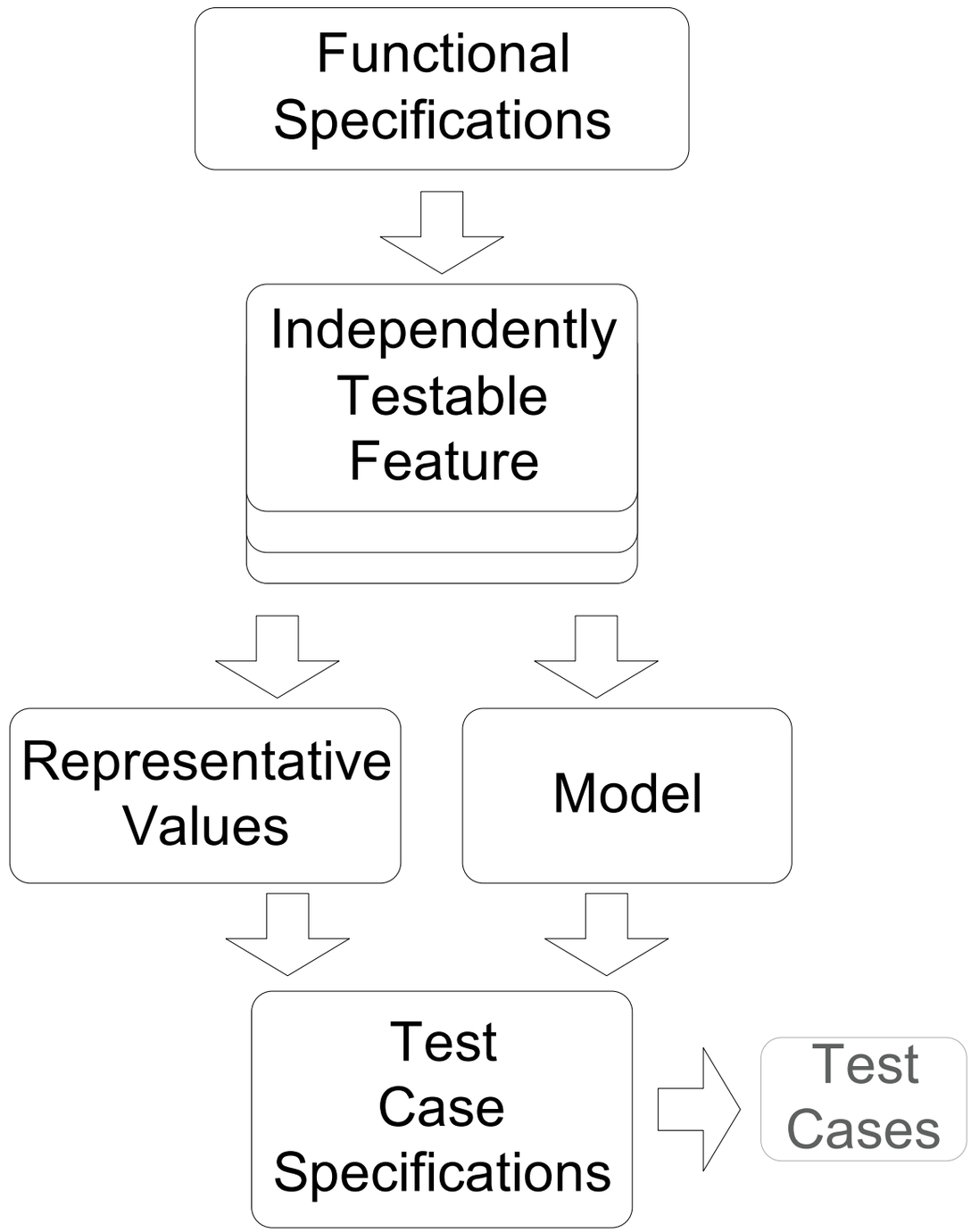
# Finite Functional Models

SWE 261P

# Recall from the past...



# Forms of Functional Models

- Diagrams (e.g., sequence, interaction, finite-state machines)
- Formal Logic
- Dynamic Invariants
- Statistical descriptions
- …

# Properties of Models

- **Compact**
  - Representable and manipulable in a reasonably compact form
  - Understandable, and thus checkable
  - What is *reasonably compact* depends largely on how the model will be used

balance betwwen these two

- **Predictive**

what project should be

  - Must represent some salient characteristics of the modeled artifact well enough to distinguish between *good* and *bad* outcomes of analysis
  - No single model represents all characteristics well enough to be useful for all kinds of analysis

# Finite State Machines

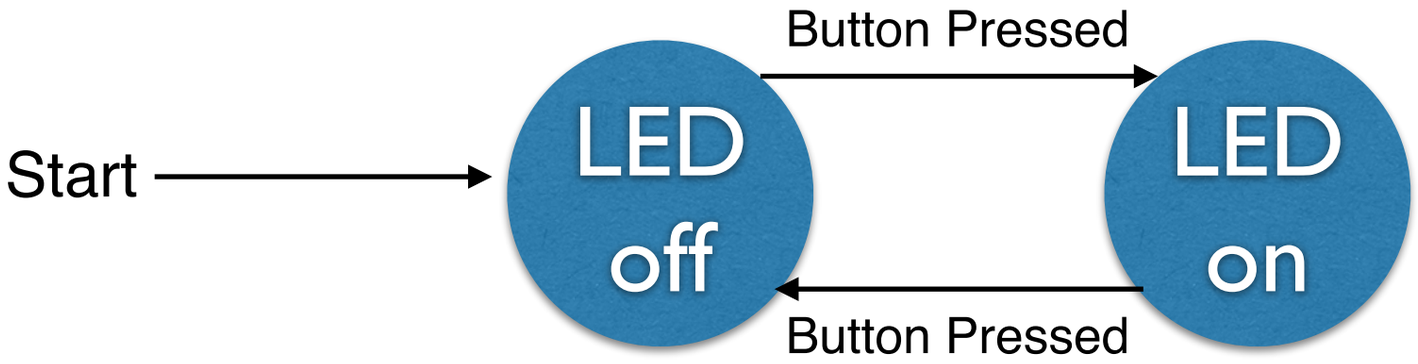
independent of code

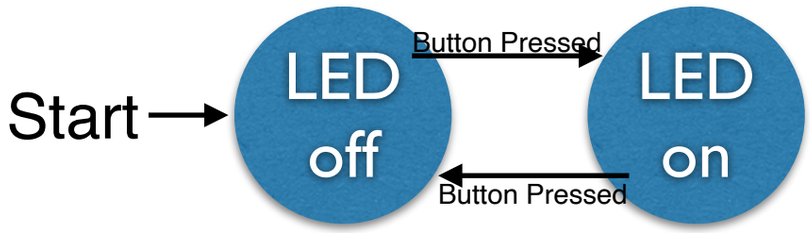
- Finite state machines (or ***FSM***s) can be constructed prior to or independent of source code.
- Can serve as a specification of allowed behavior.

# Finite State Machines

- A finite state machine is a set of states and a set of transitions.
- A directed graph.
  - Node represents a program state.
  - Edge represents an operation that transforms one program state to another. Usually are labeled with a program *operation*, *condition*, or *event*.
  - Since infinitely many states, an FSM must be an abstraction.

# Finite State Machine Example



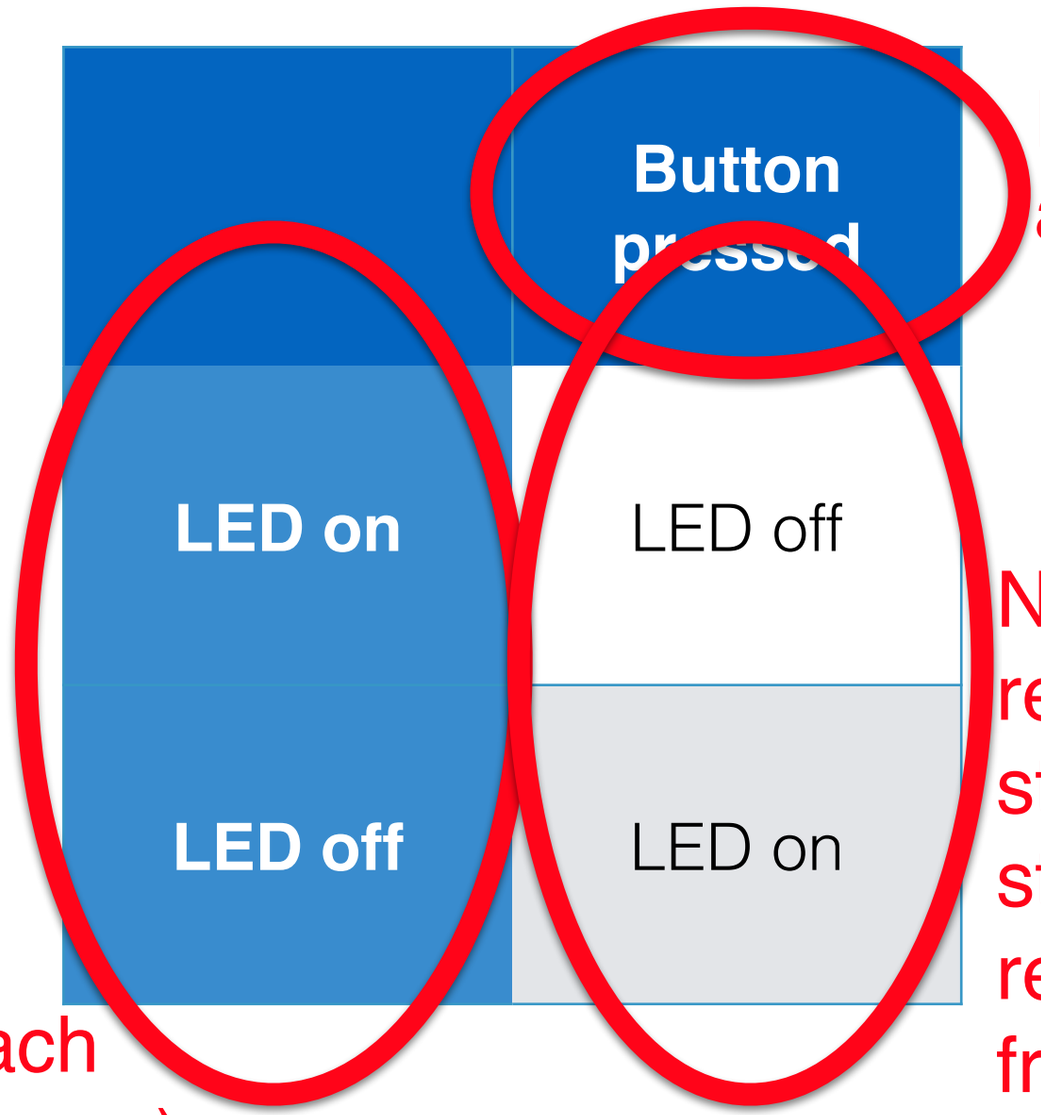



```
#define startState 0
#define ledOnState 1
#define ledOffState 2
#define buttonPressed 1
#define buttonNotPressed 0

int state = startState;
int input = buttonNotPressed;

int main(void)
{
   //init start state
   state = startState;
   while(1)
   {
      //get input first
      input = getInput();
      //
      switch(state)
      {
         case startState:
            state = ledOnState;  break;
        case ledOnState:
          if(input == buttonPressed)
           {
              state = ledOffState;
              ledOff();
           }
           break;
        case ledOffState:
           if(input == buttonPressed)
           {
              state = ledOnState;
              ledOn();
           }
           break;
        default:   state = startState;   //if unknown state, reboot
           break;
      }
   }
}
```

# Checking for Completeness: State Transition Tables




Inputs (each input in a column)

Next State (the resulting state from starting in the "current state" row upon receiving the "input" from this column)

Current State (each node/state is in a row)

# Mealy Machine

- A finite state machine that in addition to the *events*, also have a *response*.

# Example Specification

A vending machine offers one pack of gum in exchange for 10 cents. The vending machine accepts nickels (5 cents) and dimes (10 cents) only. When a nickel or dime is inserted, the machine shall create an audible beep to indicate to the customer that the machine has accepted the coin. When more than 10 cents is inserted the amount over 10 cents is returned to the customer. If any other coin is inserted, that coin is returned to the customer. When the machine has accepted 10 cents, the customer may press the "Gum" button, at which point the machine will dispense one pack of gum to the customer in exchange for the accepted 10 cents. At any point, the customer may press the "Return" button, and the machine will return the balance that the customer inserted.

# Why?

# Why?

- Capture specs
- Inform what test cases can and should be written
- Inform a “criterion” for what is “enough”

# Exercise Example #2

- Sofirn IF23 Flashlight: https://www.sofirnlight.com/products/sofirn-if23-mini-flashlight-4000lm

# Example #3

***Specification:***

A ceiling fan has two pull cords attached to it and one power button that is mounted on the wall by the door to the room. The power button acts as a toggle for activating (and deactivating) the fan — we will call this the "power button". The pull cords control the fan speed and the light bulb. The "light cord" activates and deactivates the light bulb on the ceiling fan. The "speed cord" cycles the fan through four speed settings: "1: low speed", "2: medium speed", "3: high speed", and "0: off". When the power button is already on, and the power button is then pressed, all power to the fan is effectively shut off, regardless of whether its light is on or its fan is spinning. However, for convenience of use, when the power button is already off, and the power button is then pressed, the power to the fan is resumed in such a way as to make the light bulb turn on and the fan speed is resumed to the speed that it was running prior to shutting off the power button. When first installed, the power button is in the off position (i.e., "unpowered"), and the first time the power button is pressed, the fan speed is "off" and the light-bulb is "on".