Chapter 6 – System Design II: Behavioral Models

1. Why is it important for a model to separate the design of a system from its realization? [R]

Separating the design from its realization allows the designer to see the forest instead of the trees. That is, working with models which abstract away the details of the implementation allows a designer see the entire system as a whole with being distracted by the individual parts. Furthermore, it allows the design to change the way the components are implemented. Thus, if a better technology emerges during the course of the design, the implementation of the component can be changed without affecting the overall design.

2. Classify each of the following as either, a model, not a model, or sometimes a model (explain the conditions under which this distinction takes place). [A]

a. **A diagram of a subway system.** A diagram of a subway system is a model because it is an abstract simplification of the physical subsystem. It's abstract because it's not the actual subway system; it's just a printed piece of paper. It's a simplification because it removes lots of unnecessary details in order to facilitate communication. For example, most subway diagrams are not to scale, nor do they show the actual orientation of the railways.

b. **A computer program.** A program is sometimes a model. A program modeling a physical process like a gravitational attraction of a three body system is an abstract simplification of a three interstellar bodies floating around in space. On the other hand, a program which sorts number is not necessarily a model of anything; it's just used to sort number.

c. **A football play.** A football play is a model. It's an abstraction simplification of the physical system. It's abstract because its written on paper, the X's and O's represent actual football players.

d. **A driver's license.** A driver's license is not a model. The information contained on the license describes a person and so to some extent is an abstraction of a person. However, its not the intent of the driver's license to be a model of a person, rather its intention is that it contain enough data make a unique association between a person and a privilege.

e. **A floor plan of the local shopping mall (a "you are here" diagram).** A floor plan is a model because it's a simplification of the actual shopping complex, its just a print out on paper. It's an abstraction because many of the real details of the shopping mall are not shown, like the doorways, lights, and the ubiquitous mall water fountain.

f. **An equation.** An equation is sometimes a model. The Newtonian equation representing the position of a falling body in a constant gravitational field is an abstract representation of a real falling body. It's a simplification because it neglects things like air resistance. However, an equation of a parabola is not necessarily an abstract representation of anything, it's just a parabola.

g. **A scratch n' sniff perfume advertisement in a fashion magazine**. Is a model because it captures the essential characteristic of perfume (its smell) without specifying how the perfume will produced.

h. **The 1812 overture**. Tchaikovsky (a Russian) was commissioned to write this piece of music to commemorate Napoleon's unsuccessful 1812 invasion into Russia. The musical score follows the progression of this conflict at times even calling for live cannon fire. So we could call the 1812 overture a simplified representation of the Napoleonic invasion and hence it is a model.

i. **A Braille sign reading "second floor".** It not a model for two reasons. While Braille forms representations for letters, the symbol string isn't a simplification because each Braille symbol has a one-to-one correspondence to a letter. The information contained in the message is not an effective simplification of the building's structure; we can only know that the building has at least two floors.

j. **Sheet music for the Brandenburg Concerto.** Sheet music in almost any form is a model because it's a simplification of an actual musical performance written in a standardized language.

k. **The United States Constitution.** The US constitution is a model of a government. It's a simplification because it does not try to capture all details of the eventual government, but rather establishes a framework for the structures and organization of a government.

l. **A set of car keys.** Car keys are not a model. They are not representative of anything and are not a simplification.

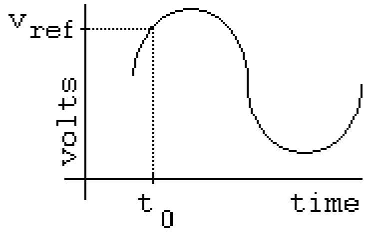
m. **The ASCII encoding of an email message.** Is not a model.

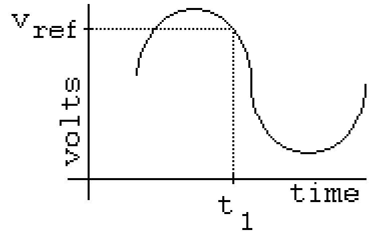
3. Which of the following systems has memory? Justify your answer using the concepts of input, output and state. [A]

a. **An ink pen.** An ink pen has memory. Let the input be the process of writing with the pen. Let the output be visible text. The same input can cause different output depending on the level of ink in the pen, empty or not empty. Sometimes a state variables (like ink level) are referred to as hidden variables, because if their value were know then the system would not appear to have state. For example if you had a see-through pen then it would be reasonable to assume that the ink level could be observed prior to writing and consequently could be considered to be an input to the system .

b. **A resistor.** A resistor does not have memory. Let the input to the resistor be current and let the output be voltage. A given current always produces the same voltage drop across the resistor, regardless of what has happen in the past.

c. **A capacitor.** A capacitor has memory. Let the input be voltage and the output current. The relationship between these two is characterized by the equation, I=Cdv/dt. The graphs below represent a time varying voltage applied across a capacitor. Note that the voltage measured at time t_0 and t_1 are the same, v_{ref}. However, the current through the cap will be different at times t_0 and t_1 because the slope of the voltage vs. time graph is different.





d. **A motorized garage door.** A motorized garage door has memory. Let the input be the button press to activate door movement, the output be the movement of the door, and the state variable be the position of the door. When the button is pressed the direction of the door's movement (opening or closing) depends on weather the door was open or closed.

e. **An analog wrist watch.** An analog wrist watch has memory. The input is time and the output is the indicated time. The output is determine by the passage of time and the value of the previous time.

f. **The air pressure in an air compressor.** The air pressure itself does not have memory, but the air compressor does have memory. Compressors have a high and low pressure threshold. When the pressure in the compressor tank falls below the low threshold, the air pump automatically turns on, increasing the pressure in the tank. When the pressure in the tank exceeds the high pressure threshold the compressor turns off. When the pressure is between the two thresholds the compressor keeps doing what it had been doing moments before.

g. **The thermostat which controls the furnace in a house**. The thermostat to control your house furnace has memory. When you set the temperature of a thermostat you are really setting two temperatures, a low threshold and a high threshold. When the temperature falls below the low threshold the furnace turns on heating the house. When the temperature rises below the high threshold the furnace turns off cooling the house. When the temperature is between the two thresholds the furnace keeps doing what it had be doing moments before. The input is the temperature, the output the state of the furnace, and the state variable the previous state of the furnace.

h. **Light switch.** A common 1-way light switch doesn't have state – the position of the switch determines the state of the light. A 2-way switch does have state. 2-way switches are used in homes to control a stairwell light from either the top or bottom of the stairs. Let the input be the position of the light switch at the bottom of the stairs, the output be the state of the light bulb, and the state variable be the position of the light switch at the top of the stairs.

i. **A political system.** Political systems should have memory. For example, a bill to fund the local puppy shelter is not passed. Irate voters take their let their representatives know about their displeasure. The same bill coming up a year later is more likely to pass.

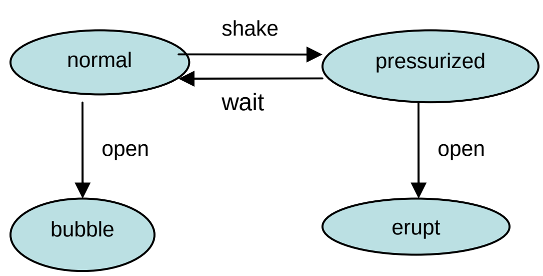
j. **The temperature of a large lake.** The temperature of a large lake has memory. Let the input be the amount of light falling on a lake and the output be the temperature of the lake. In the northern United States lakes will reach their peak temperature in late August or early September whereas sunlight intensity peaks on the summer solstice (around June 21). Even the surface air temperatures peak in late July or early August, about a full month ahead of the peak lake temperatures.

k. **A book.** Books don't have memory.

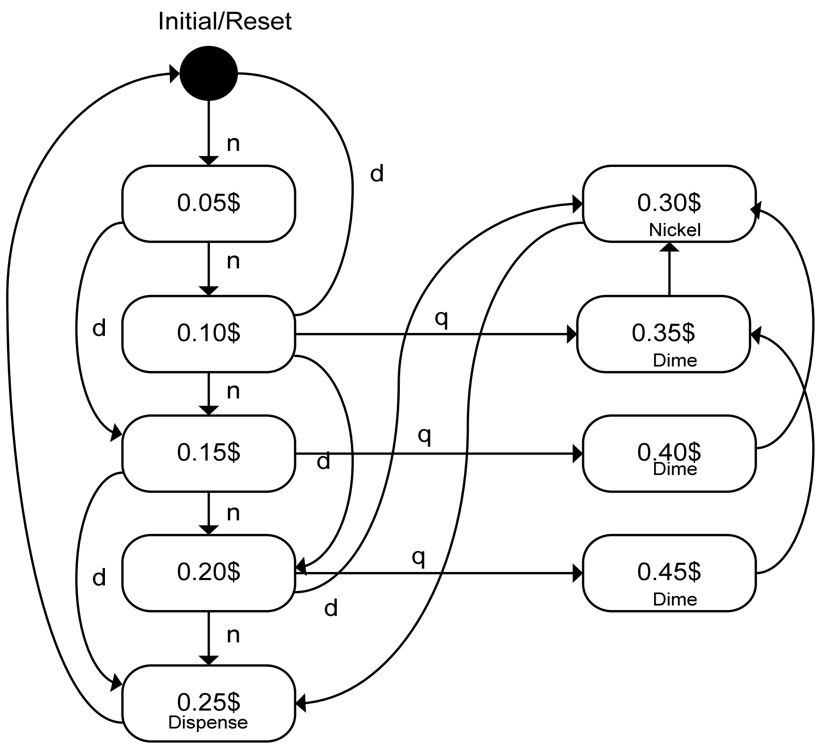
l. **A computer's hard drive.** A computer hard drive has memory because a byte read from a location depends on what has been written to the drive in the past.

4. **A can of soda has memory. Your objective is to figure out what characteristic of the can is the state variable and what input causes it to change. Based upon this, draw a state diagram for a can of soda. Label the transition arcs with the input responsible for the transition. Hint: no special equipment is needed to elicit the change of state. [A]**

We will use the "internal pressure" of the can as the state variable. When you shake a can of soda vigorously you "pressurize" it. In order to decrease the pressure, all you need to do is wait. The can of soda has state because it remembers previous input; did you recently shake the can? Opening the can will reveal which of the two states it is in.



5. **Consider the state diagram for the vending machine shown in Figure 6.2. Now assume that the system accepts nickels, dimes, and quarters. Also assume that it is capable of returning change to the user after a purchase. Create a state diagram that represents this new system. Make sure to define the output signals and their value for each state. [A]**



The state diagram for the modified vending machine contains states for all monetary combinations from \$0.00 to \$0.45; that is all states that the vending machine can be placed in before a soda is dispensed. All the states except \$0.25 the vending machine outputs a soda and returns to the reset state. In state with a monetary value above \$0.25 the vending machine dispenses change. This output behavior is captured in the output table shown below.

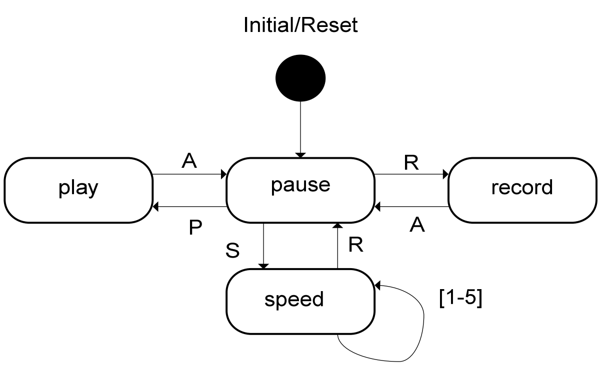
State	Dispense	Nickel change	Dime change
Initial/Reset			
\$0.05			
\$0.10			
\$0.15			
\$0.20			
\$0.25	Yes		
\$0.30		Yes	
\$0.35			Yes
\$0.40			Yes
\$0.45			Yes

6. **Draw a state diagram to describe the high level operation of the ChipMunk Recorder (CMR). The CMR records sounds and then plays them back at a variety of speeds, making a recorded voice sound like a high pitched chipmunk.**

The CMR receives user input from a keyboard and an audio source. The behavior of the system is described as follows:

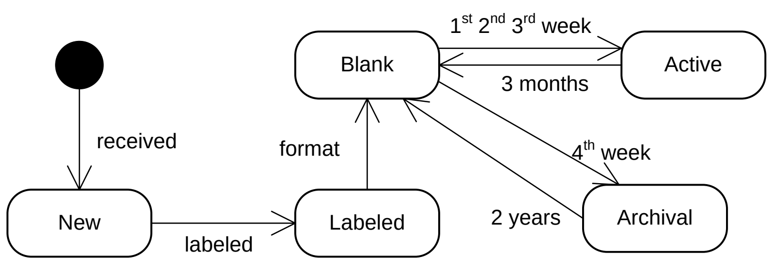
> **When powered up, the CMR enters a wait state.**
>
> **If 'R' is pressed the recorder begins recording.**
>
> **Any key-press will put the CMR back into the wait state.**
>
> **If 'S' is pressed, the CMR is ready to change the playback speed. A subsequent numerical input between 1 and 5 will cause the playback speed to be changed to that value.**
>
> **Pressing the 'R' key when in the adjust playback speed mode will cause the CMR to go to the wait state.**
>
> **Pressing a 'P' key will cause the CMR to playback the recorded sounds. When done playing the entire recording, the CMR will loop back and start playing at the beginning.**
>
> **Any key-press while in the playback mode will cause the CMR to go back into the wait state.**

Draw a state diagram describing the behavior of the CMR. Create a table which lists every state and its associated output. [A]



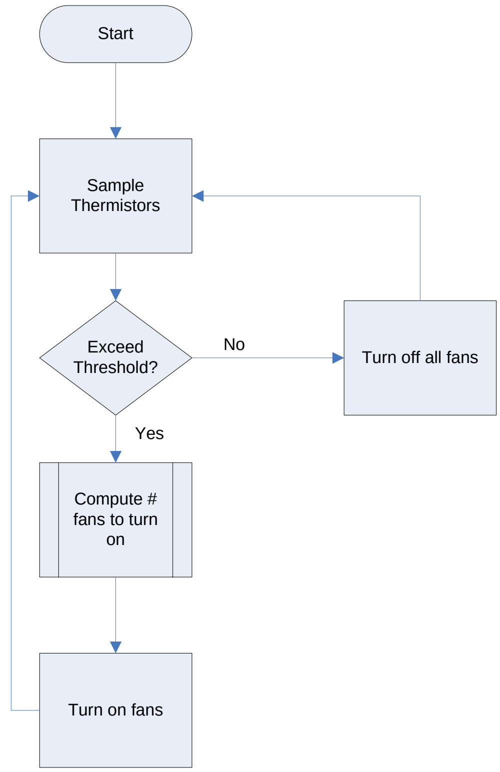
State	Output
pause	All subsystems off
record	Enable record subsystem
speed	Change speed (and store)
play	Play at speed with loop-back to beginning at end

7. **Build a state diagram to describe the state of the tape cartridges used to backup a company's network drives. When new unformatted cartridges are received they are immediately labeled with a unique ID. Before a tape is used it is formatted turning it into a blank tape. On the first day of the week a complete backup is made of the network drives, transforming blank tapes into active tapes. The active tapes, made every fourth week, are moved off site making them archival tapes. Active tapes older than 3 months are assumed to have out of date information and are reformatted into blank tapes. Archival tapes more than 2 years old are reformatted and put back into circulation.**



8. **Build a flowchart to describe the operation of a microcontroller (MCU) based temperature regulating system. The system monitors the temperature of a heated environment using thermistors and regulates the temperature by turning on fans. The MCU periodically reads the temperature from 64 different thermistors (each is driven by its own constant current source) by selecting each through an analog multiplexer. The resulting voltage is converted into an 8-bit digital value using the MCU's analog-to-digital converter. There are two temperature ranges recognized—low temperature and working temperature. When the temperature is low, the MCU turns all the fans off. When the temperature is in the working range, the MCU uses a complex algorithm to determine the number of fans to turn on. [A]**

   This solution to this problem shows how a model abstract away details. For example, there is no need to get involved with constant current sources in the flowchart. Additionally, the solution presented does not even bother to describe the number of thermistors used; rather, it makes do with a "Sample Thermistors" step. The flowchart captures the essential features of the cooling process by showing that a decision is made to turn the fans on or off depending on the value of the thermistors. Certainly there is more than one correct solution to this problem, however the designer should strive for a balance between keeping the number of operations to a minimum while still capturing the iterative nature of the process.

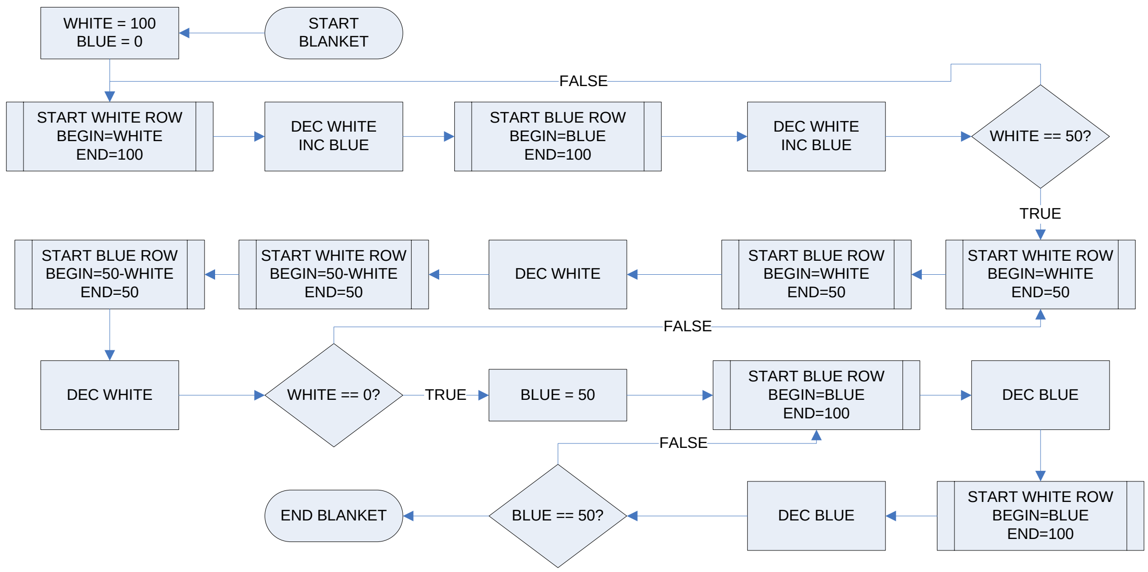

Start
Sample Thermistors
Exceed Threshold?
No
Turn off all fans
Yes
Compute # fans to turn on
Turn on fans

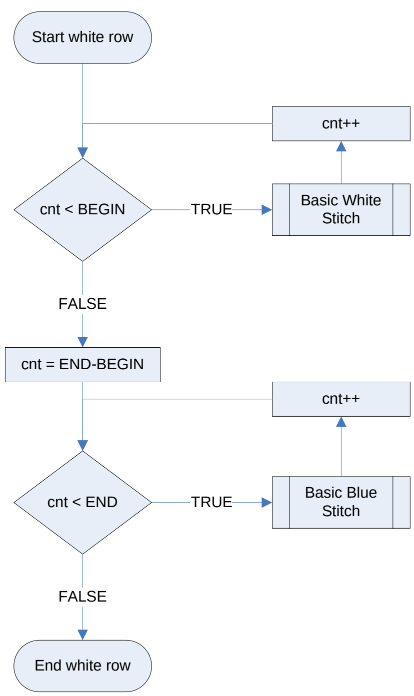
9. Write an algorithmic description for each of the flow charts below using while, if, or do statements.

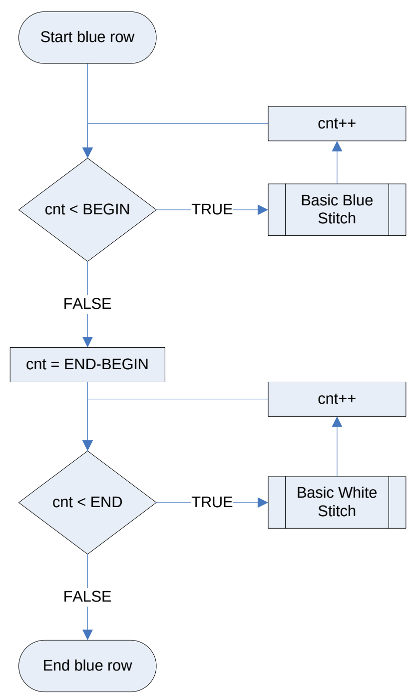
```
a. while(x) y;
b. do y while(x);
c. if (a) {if (b) s1 else s2} else {if (c) s3 else s4}
```

10. Create a flowchart that outlines how to crochet a two-tone blanket with a diagonal stripe across it as shown below. A blanket is crocheted by linking together a sequence of basic stitches. For the purposes of the flowchart assume that a basic stitch is an elaborate process. Basic stitches are made from either dark or light yarn. The blanket should be 100 stitches wide by 150 stitches high. The diagonal stripe runs at a 45 degree angle from the horizontal.

This process starts at the upper left corner of the blanket and proceeds to the right forming one row of 100 basic stitches and then works back to the left forming the second row. This process repeats for the next 148 rows.

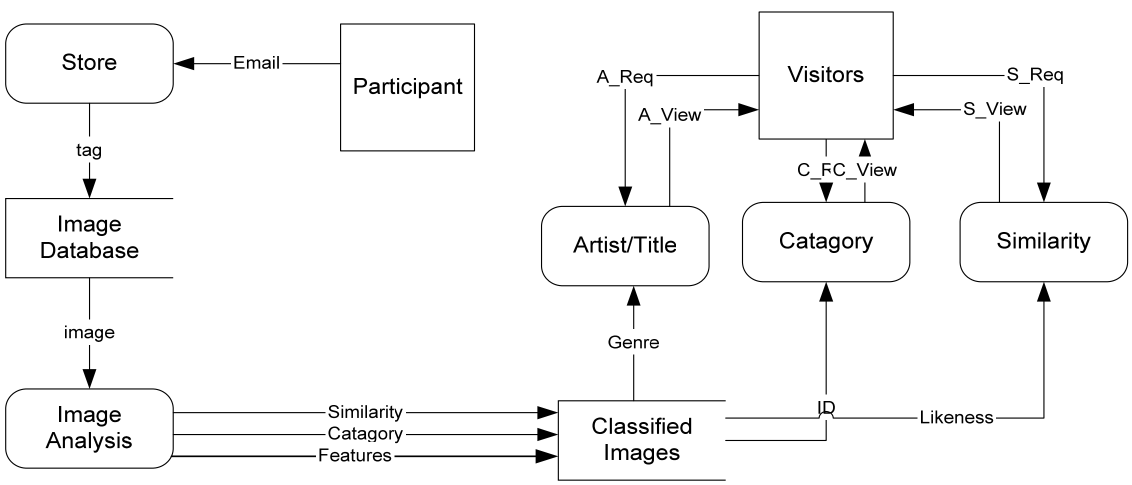

WHITE = 100
BLUE = 0
START
BLANKET
FALSE
START WHITE ROW
BEGIN=WHITE
END=100
DEC WHITE
INC BLUE
START BLUE ROW
BEGIN=BLUE
END=100
DEC WHITE
INC BLUE
WHITE == 50?
TRUE
START BLUE ROW
BEGIN=50-WHITE
END=50
START WHITE ROW
BEGIN=50-WHITE
END=50
DEC WHITE
START BLUE ROW
BEGIN=WHITE
END=50
START WHITE ROW
BEGIN=WHITE
END=50
FALSE
DEC WHITE
WHITE == 0?
TRUE
BLUE = 50
START BLUE ROW
BEGIN=BLUE
END=100
DEC BLUE
FALSE
END BLANKET
BLUE == 50?
DEC BLUE
START WHITE ROW
BEGIN=BLUE
END=100


Start white row
cnt++
cnt < BEGIN
TRUE
Basic White
Stitch
FALSE
cnt = END-BEGIN
cnt++
cnt < END
TRUE
Basic Blue
Stitch
FALSE
End white row


Start blue row
cnt++
cnt < BEGIN
TRUE
Basic Blue
Stitch
FALSE
cnt = END-BEGIN
cnt++
cnt < END
TRUE
Basic White
Stitch
FALSE
End blue row

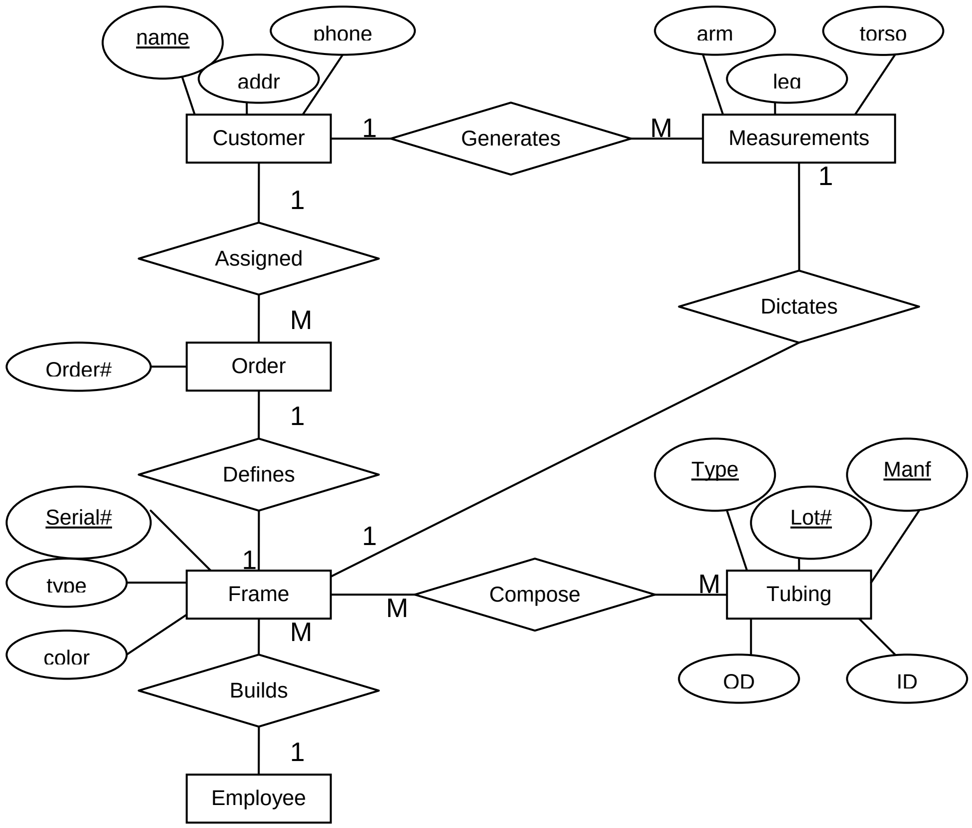
11. Build a data flow diagram and event table to represent an image archiving system for an art museum. The art museum maintains a database of digital images of paintings from museums all over the world. The following information is known about the image database system: [A]

- Images are shared among museums in a participating network. Whenever a participating museum posts a new image, it sends a broadcast email to all participating museums in the network with the image attached as an email. It provides the name of the painting and artist in the body of the email. All new images received are added directly to the museum's own image database.
- When inserted into the museum's database, each image is provided with a tag identifying the name of the painting and the artist. Furthermore, this triggers an image analysis routine that classifies the image based into predefined categories such as portrait, natural scene, and modern. Furthermore, it stores key features that are extracted from the image.
- The key features are used to identify and retrieve visually similar images from the museum's database. Another image processing algorithm is run that compares the visual similarity of the new image to all images in the database. This process produces a matching score of 0 to 100 that is stored.
- The museum's image database is available to visitors via computer kiosks placed throughout the museum. Kiosk users can retrieve and view images in one of three ways. First, they can specify the name of the artist or painting. Second, they can retrieve a class of images, such as modern. Third, once they have received a painting, they can submit a request to view visually similar images. The visually similar images are retrieved for viewing based upon the matching score.

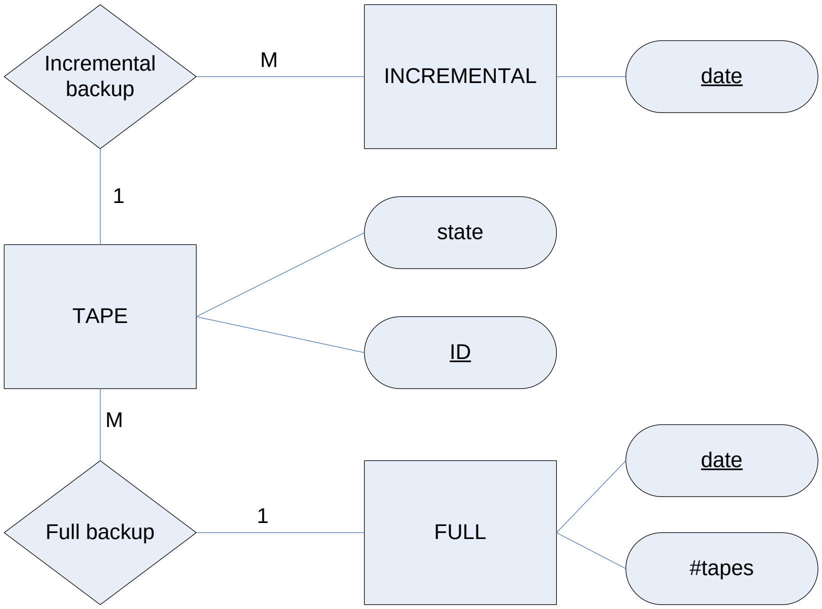


12. **Build an ERD to keep track of the bicycle frames manufactured at a local company. [A]**

In creating an ERD its typical to “grow” the ERD from a single entity. As a starting point, lets consider the frame entity. Clearly, a serial number uniquely identifies a particular frame. The type and color of the frame are attributes. Its a common mistake for students to list the possible values of the frame type as attributes of a frame. This is incorrect, an attribute represents the domain of possible values. Is the tubing used to build a frame an attribute or an entity itself? Anytime that you can generate a unique identifier for a thing then its a good bet that it should be its own entity. In the case of tubing, we can combine the lot#, manufacture, and type to distinguish between tubes. We assume that a bike frame is made from 1 type of tubing and a particular lot of tubes can be used to build many frames. One might be tempted to relate customers and frames, however, there is nothing in the word statement that explicitly requires this. Its better to relate frames to the orders which generate them. We can then relate customers to their orders. The cardinality of these two relationships are clear from the diagram. From the notes gathered its clear that we need to store the customer’s name, address, and phone number. These are all attributes which help to identify a particular customer. We could use the customers name as the primary key, it’s a safe bet that a particular name will not be repeated. Finally we need to relate the customer to their measurements. Since the word statement states that a customers measurements may change over time, we cannot have a measurement attribute, because an attribute is only allowed to take on a single value for a particular entity instance. Hence, we need to make a measurement entity and allow each customer to have many measurements. Unfortunately, the arm, leg and torso attributes are not sufficient to uniquely identify a particular measurement, its possible for different people to have the same measurements. In ERD nomenclature, a entity which lacks a good primary key is called a weak entity. When converted into a relational database a weak entity "borrows" its primary key from the participating entities.



13. **Extend Problem 6.7 to create an ERD that captures data about the tape cartridges used in the backup system. Every Sunday night a full backup is made of all network drives. A full backup creates an identical copy of the network drives on the tape cartridges. Due to the large amount of information, a full backup requires many tapes. On the other nights of the week an incremental backup is made. An incremental backup stores only files modified since the last backup (either full or incremental). Incremental back-ups are much smaller than a full backup, and consequently many incremental backups fit on a single tape. A tape contains only full or incremental backup information – the unused portion of the last tape used for a full backup is never used to store incremental backups. Your company wants to keep track of tapes, full backups, and incremental backups. An ID and state should be tracked for each tape. For full backups, the system needs to track the creation date and the number of tapes used. For an incremental backup it should track the date it was made. The relationships between the backup type and the tape will capture which tapes participated in which backup. (Hint: the state of a tape should be an attribute of the tape entity - unformatted, blank, etc. They are not attributes and are possible values for the state attribute.)**



14. Project Application. Develop behavior models that are applicable for describing your system. Table 6.4 is provided to help in making the determination as to which models are applicable. [P]

When students define their projects using the functional decomposition method outlined in chapter 5, they typically will use block diagram listing the inputs, outputs and behaviors for the highest levels of abstraction. However, as the lower levels, the use of alternative design models becomes important to formally describe the behavior of their systems. It is not uncommon for students to use several different languages to describe their systems. For example, a group that is building an autonomous robot may use both state diagrams and flow charts to describe their algorithms.