Reading 7 -- State Diagrams x Requirements Description and Use-cases

Goal: To verify that the state diagrams describe appropriate states of objects and events that trigger state changes as described by the requirements and use cases.

Inputs to process:
1. The set of all state diagrams, each of which describes an object in the system.
2. A set of functional requirements that describes the concepts and services that are necessary in the final system.
3. The set of use cases that describe the important concepts of the system

For **each state diagram**, do the following steps:

1) **Read the state diagram to basically understand the object it is modeling.**
2) **Read the requirements description to determine the possible states of the object, which states are adjacent to each other, and events that cause the state changes.**

INPUTS:	Requirements Description (RD)
OUTPUTS:	Object States (marked in blue on SD)
	Adjacency Matrix

☞ Put away the state diagram and erase any (*) from that are in the requirements from previous iterations of this step. Now, read through the requirements looking for places where the concept is described or for any functional requirements in which the concept participates or is affected. When you locate one of these, mark it in pencil with a (*) so that it will be easier to use for the remainder of the step. Focus on these parts of the RD for the rest of the step.

☞ Locate descriptions of all of the different states that this object can be in. To locate a state, look for attribute vales or combinations of attribute values that can cause the object to behave in a different way. When you locate a state underline it with a blue pen and give it a number.

☞ Now identify which one of the numbered states is the Initial state. Using a blue pen, mark it with an “I”. Likewise mark the end state with an “E”.

☞ When you have found all of the states, on a separate sheet of paper, create a matrix with 1..N across the top and 1..N down the left side, where 1..N represents the numbers that you gave to the states in the previous step.

☞ For each pair of states, if the object can change from the state represented by the number on the left hand side to the state represented by the number on the top row, then mark the box at the intersection of the row and column. If you can determine the event(s) that cause the state change put that in the box, if not just put a check mark (the event will be determined in a later step). If you can determine that it is not possible for the transition to happen then place an X in the box. If you cannot make a definite determination then leave the box blank for now.

☞ For any event that you have identified above, if there are any constraints described in the requirements, then write those by the event in the matrix.

3) **Read the Use cases and determine the events that can cause state changes.**

INPUT:	Use Cases
OUTPUT:	Completed Adjacency Matrix

☞ Read through the use cases and find the ones in which the object participates. Focus on these for the rest of the step.

☞ For each box in the adjacency matrix that has a check mark in it, look through the use cases and determine what event(s) can cause that transition. These events may not be obvious and may require you to abstract the use-cases and think about what is actually going on with each object. Erase the check mark and write this event(s) in its place.

☞ For each box that is blank in the adjacency matrix, see if any event that can cause that transition is described in the use cases. If it is, then write that event in the box, if not then place an X in the box.

4) **Read the state diagram to determine if the states described are consistent with the requirements and if the transitions are consistent with the requirements and use cases.**

INPUT:	Requirements Description;
	State Diagram (SD);
	Adjacency Matrix (AM).
OUTPUT:	Discrepancy Reports

☞ For each state that is marked and numbered in the requirements description, find the corresponding state on the state diagram and using a blue pen, mark it with the same number used in the requirements. Be careful, because the same state may have a different name in the requirements than it has on the state diagram. To determine if two different names are talking about the same state, you must use your understanding of the requirement's description of the state and the information contained in the state diagram. This may be an iterative process where if states appear to be missing, you must go back and look again at what you have identified and make sure that it is correct.

Were you able to find all of the states?

If a state is missing, look to see if two or more states that you marked in the requirements were combined into one state on the state diagram. If not, then you have found a defect of Omission. If so, then does this combination make sense? If not, you have found a defect of Incorrect Fact.

Where there extra states in the state diagram?

Look to see if one state that you marked in the requirements has been split into two or more states in the state diagram. If not, then you have found a defect of Extraneous. If so, does this split make sense? If not, you have found a defect of Incorrect Fact.

- ☞ Once you have all of the states labeled with numbers, using the AM, compare the transition events marked on the matrix to the ones on the SD. For any box on the AM that is marked with an event, check the corresponding states on the SD to make sure they have an event to transition between them, and check to ensure that the event is the same.

Do all of the events on the AM appear on the SD? If not, you have found a defect of omission. Do events appear on the SD that are not on the AM? If so, you have found a defect of extraneous fact.

- ☞ For each constraint that was marked on the AM, find it on the SD.

Did you find all of the constraints that are on the AM? If not, then you have found a defect of omission. Did you find a constraint on the state diagram that is not on the AM? If so, does the constraint make sense? If not then you have found a defect of extraneous fact.