Architectural Design and Interfacing

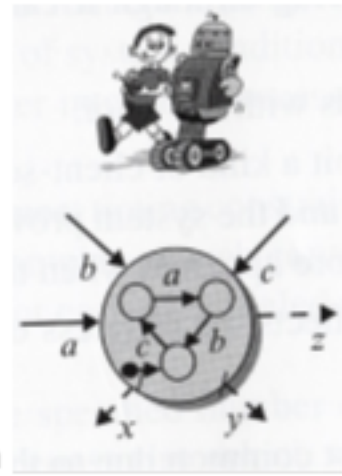
Objectives

- Define the 4 types of systems and their behavior
- Categorize systems into the appropriate classifications
- Explain the steps in object interaction modeling
- Critique examples of notation for correctness
- Construct Interaction diagrams to define the interactions between multiple classes

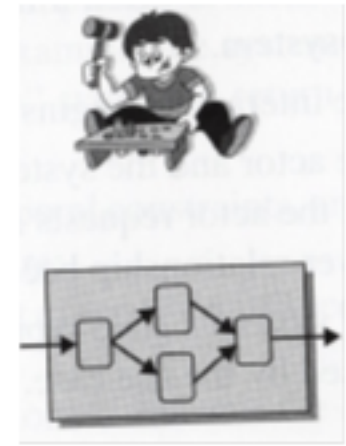
4 types of systems

- Interactive Systems
  - System directly processes interactions with the actor
  - Actor tends to be a human being
  - Interaction begins and ends with the actor
  - System's state represents progress through business activity

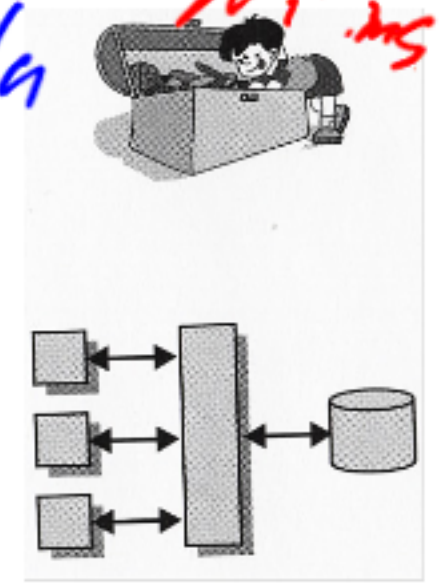
- Event Driven Systems
  - System receives events from and controls external entities
  - Inputs come in in random fashion
  - Event responses dependent upon state
  - External entities typically hardware
  - System may need to meet timing constraints



- Transformational Systems
  - Viewed as a network of information processing activities
  - Little or no interaction between the system and the actor
  - Usually stateless systems
  - Number crunching and computationally intensive



- Database systems
  - Data is shielded from changes to the rest of the system
  - Responsible for only storing and retrieving data from a system
  - Capable of efficient storage and retrieval of a huge amount of data.

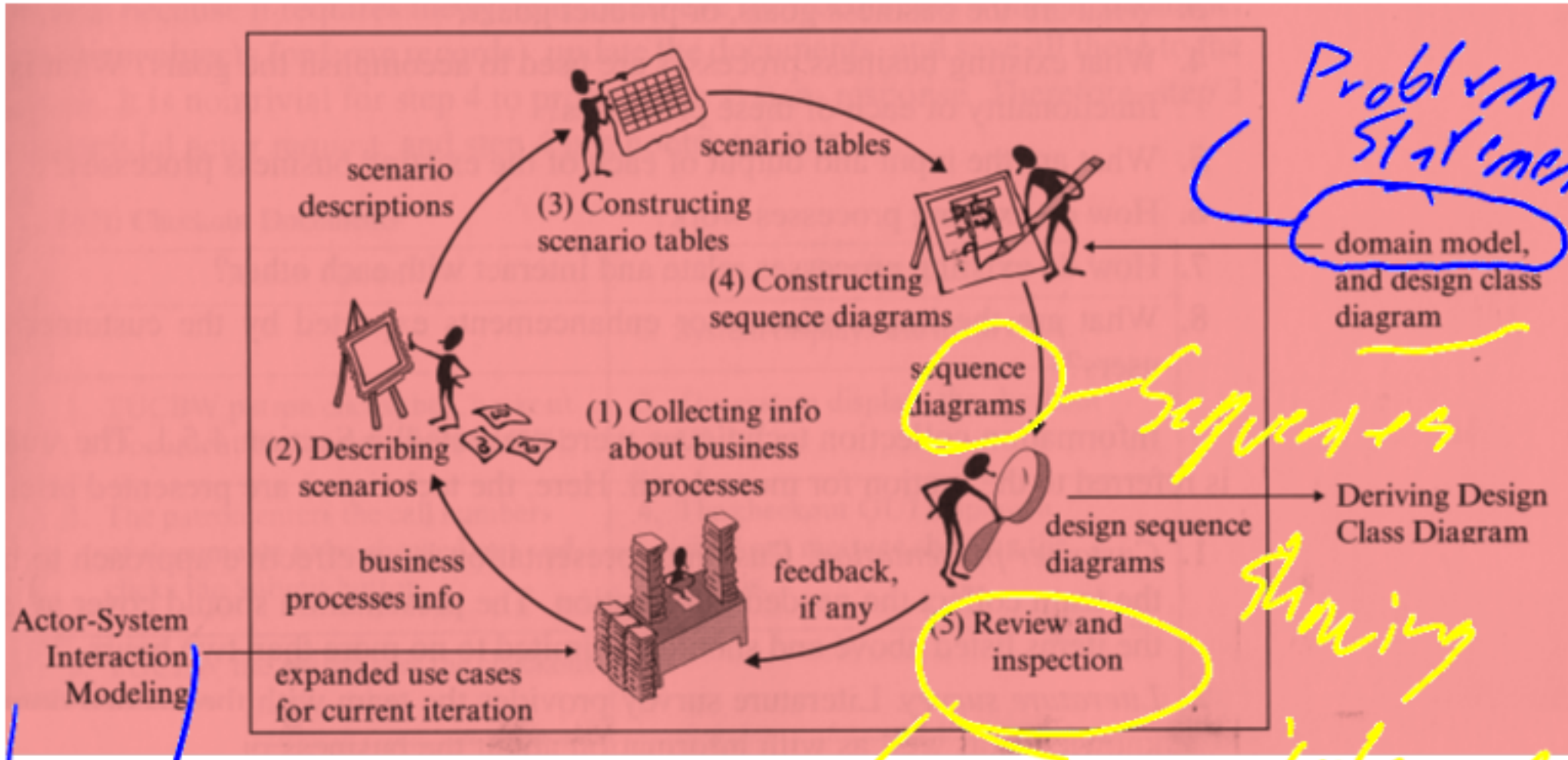


Partner Exercise

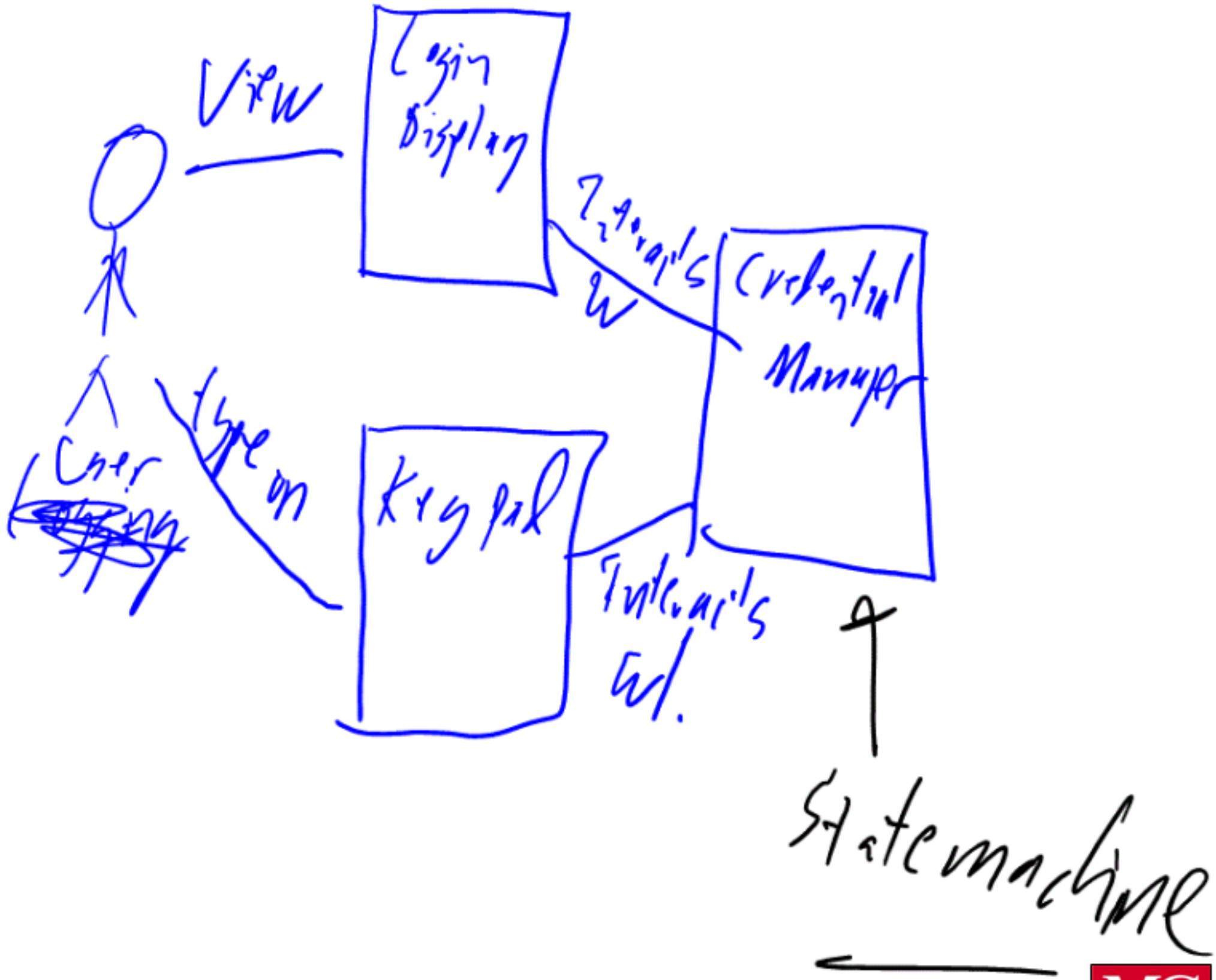
- With the partner next to you, come up with an example of each type of system
  - Interactive System
  - Event Driven Subsystem
  - Transformational System
  - Database System

Identifying interactions :

The process



What objects exist in a login system?



- On a login system, what states may exist?

Login System

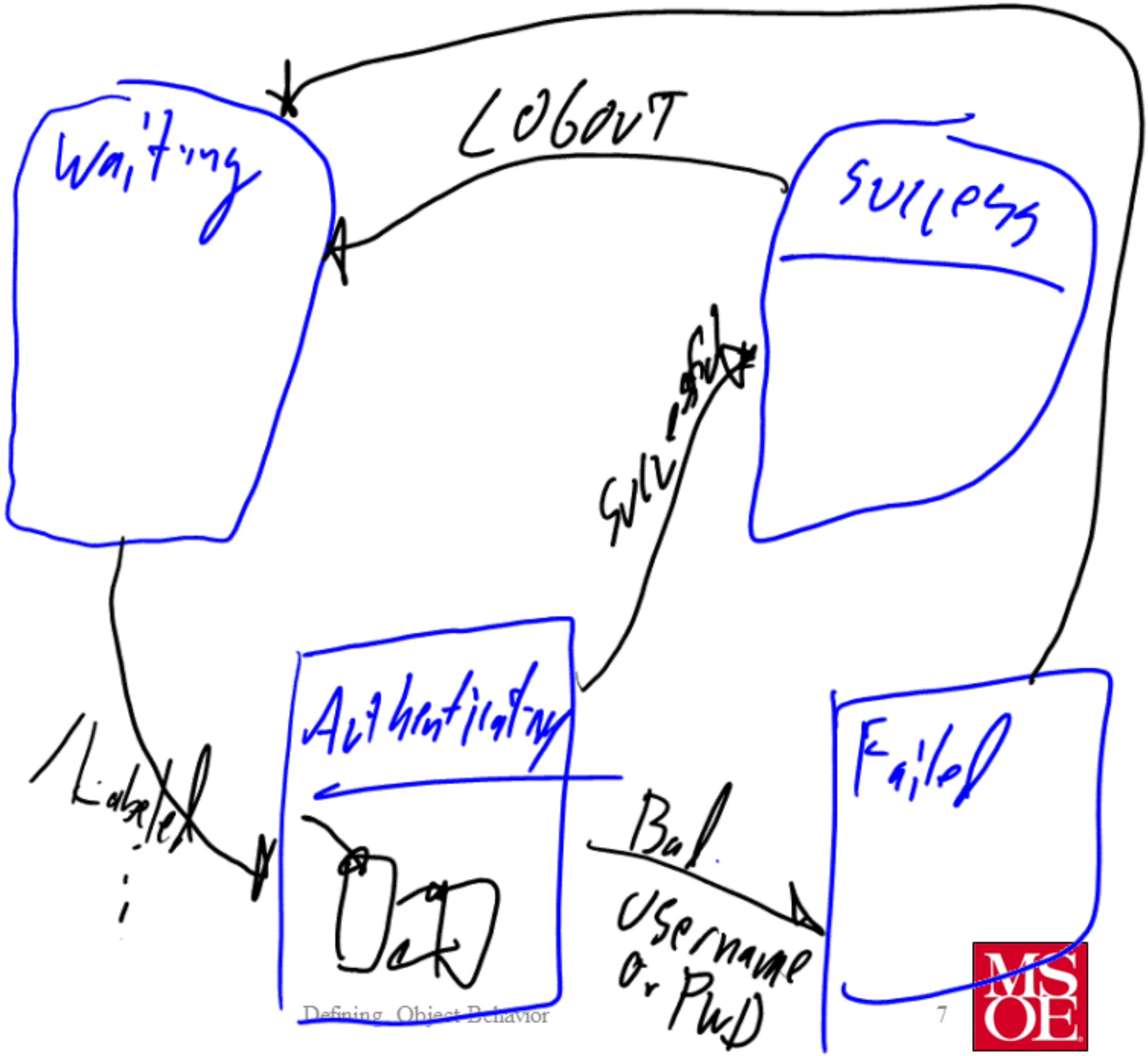
Waiting

Authenticating

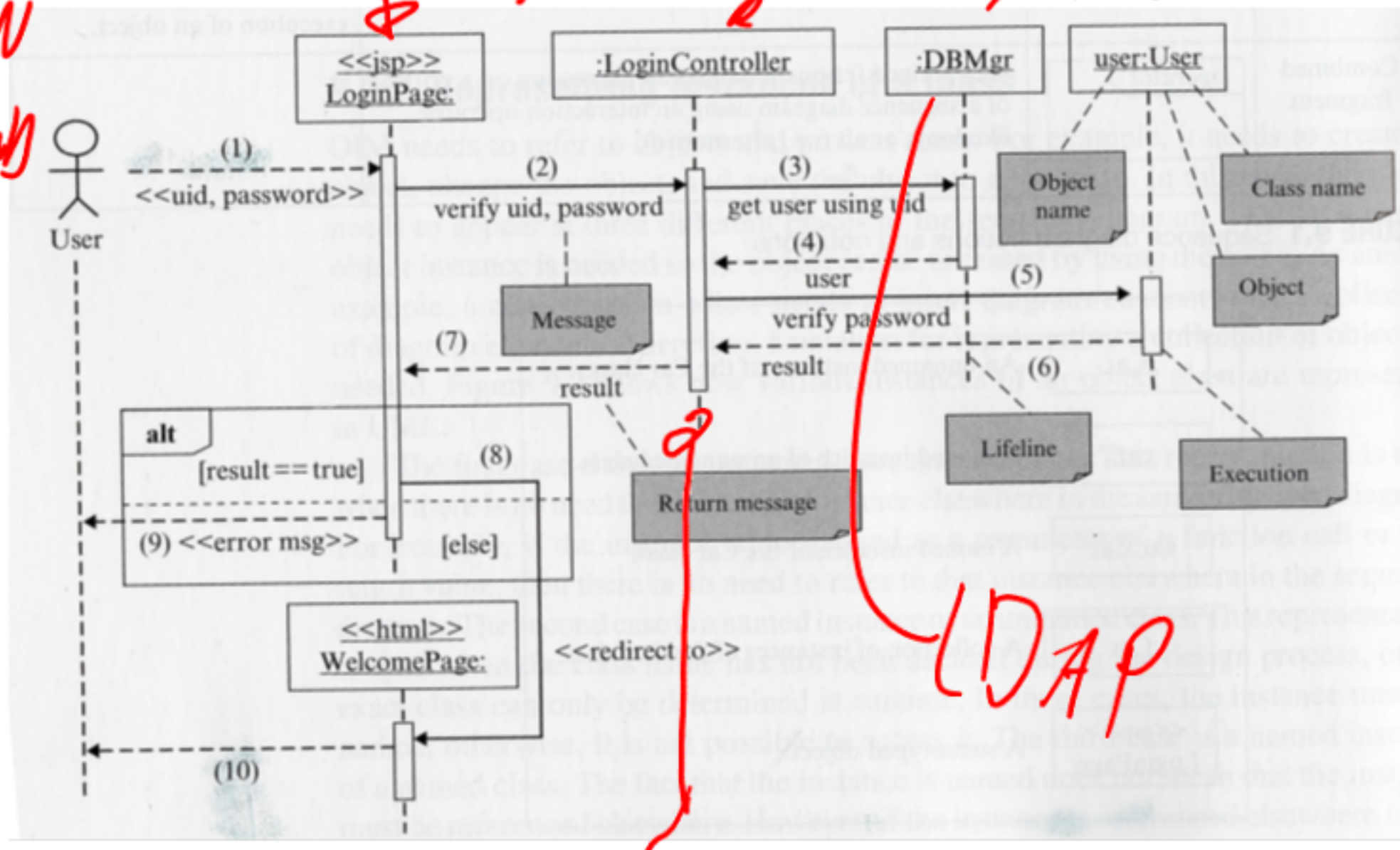
Failed

Authenticated = Success

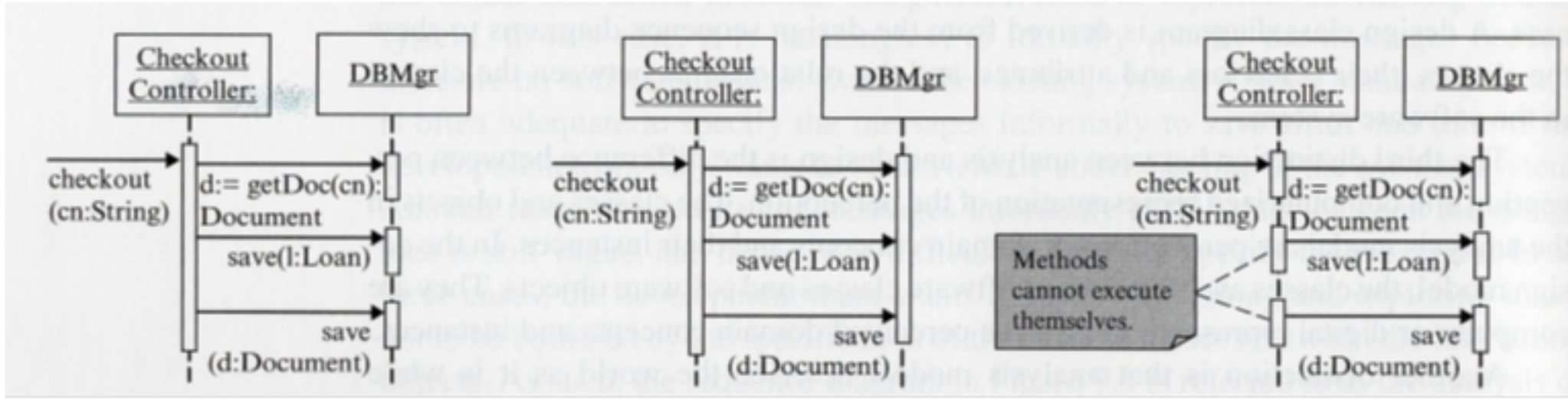
Login System State diagram



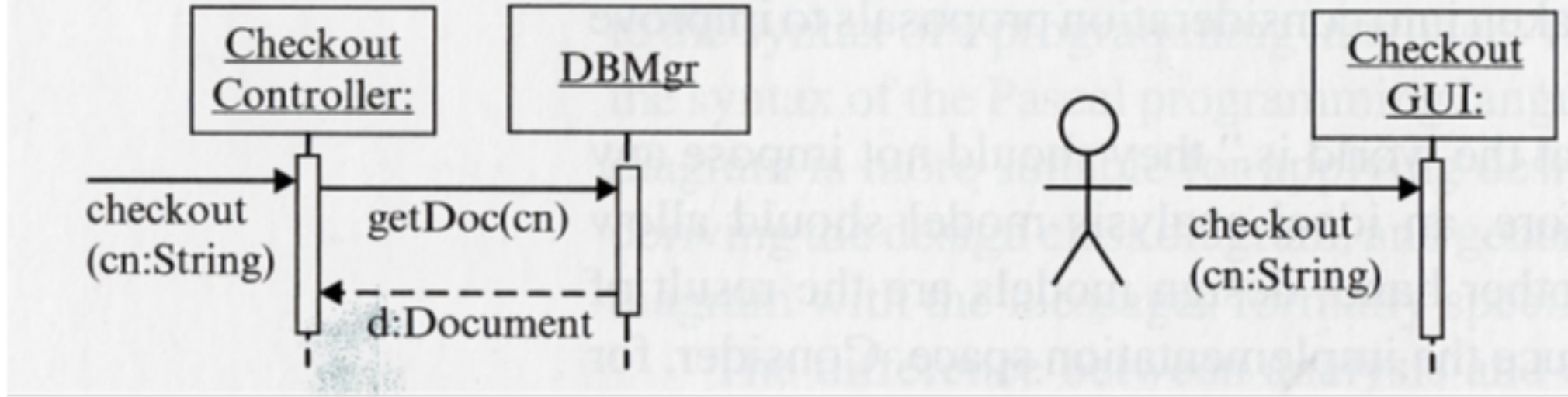
Login Sequence Diagram



UML Sequence Diagrams: Which is right?



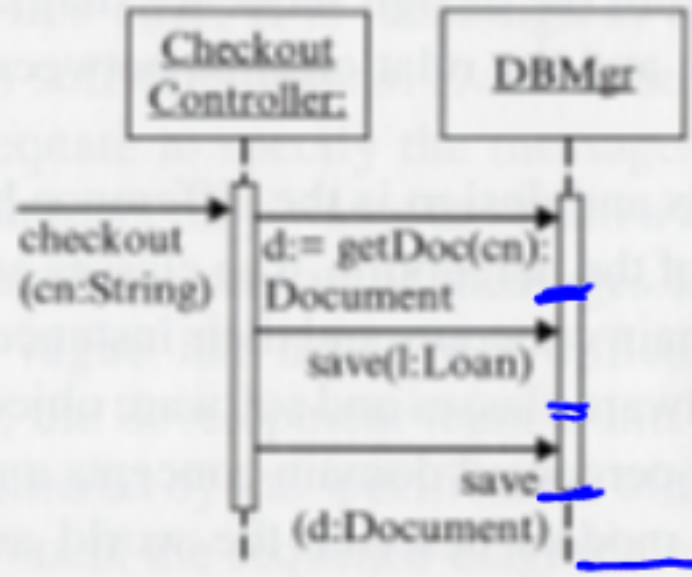
A B C



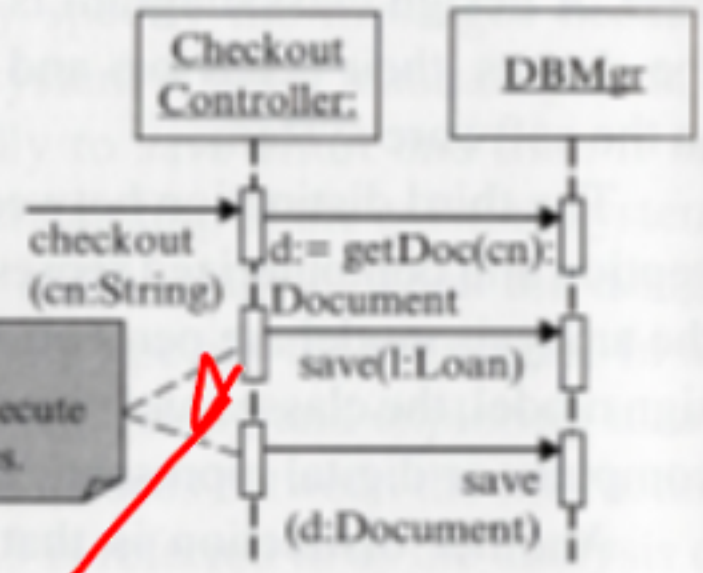
D -3 E

Abstains

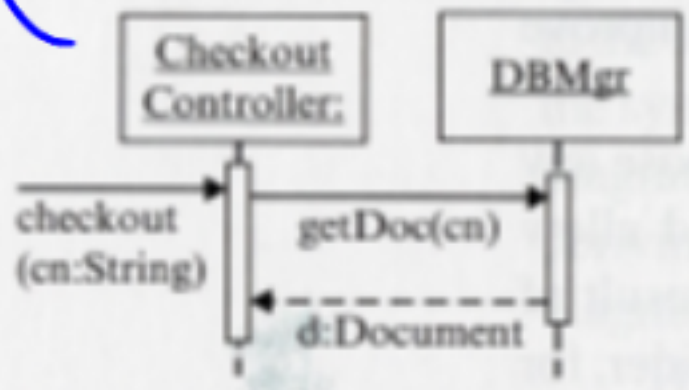
(a) Correct: during the execution of checkout(...), three separate calls to DBMgr are made.



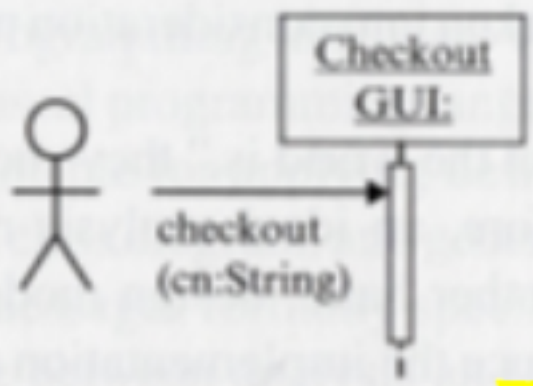
(b) Incorrect: the long rectangle beneath DBMgr should split into three as in (a).



(c) Incorrect: methods must be called to execute.

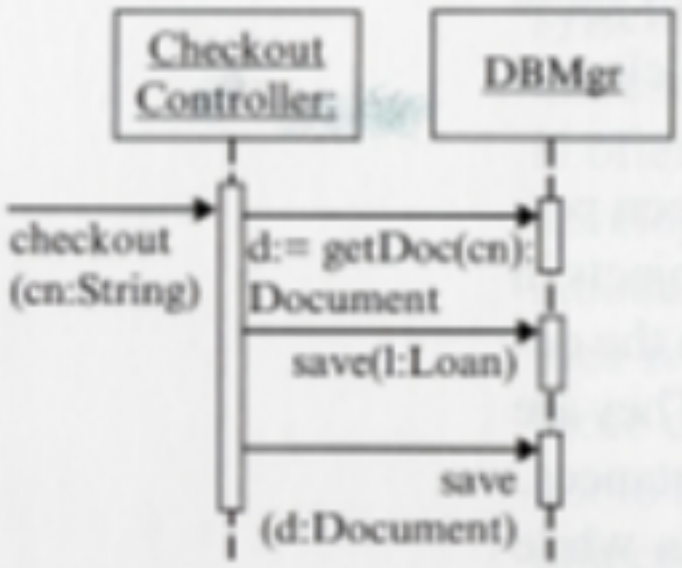


(d) Not preferred: the back dashed arrow line can be interpreted differently. Do as (a) is preferred.

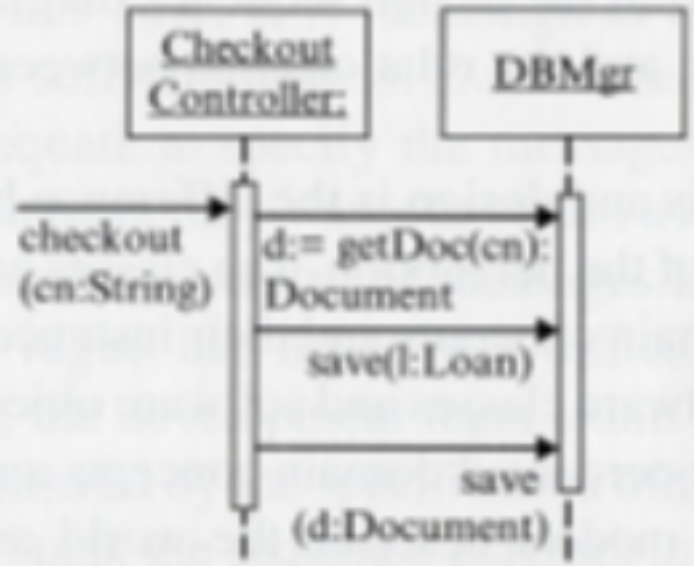


(e) Incorrect: an actor cannot call a function of an object; should use a dashed line and stereotype message.

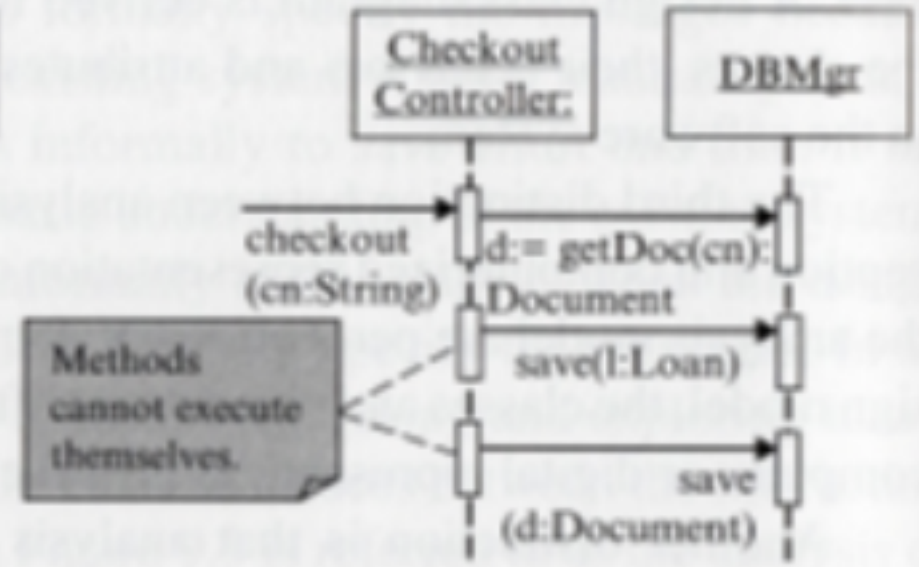
FIGURE 9.6 Correct and incorrect uses of notations



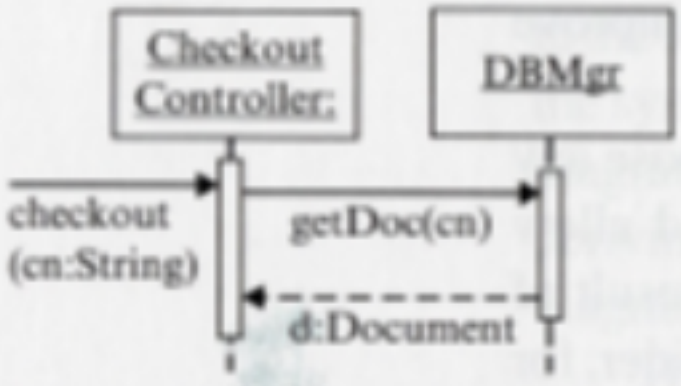
(a) Correct: during the execution of checkout(...), three separate calls to DBMgr are made.



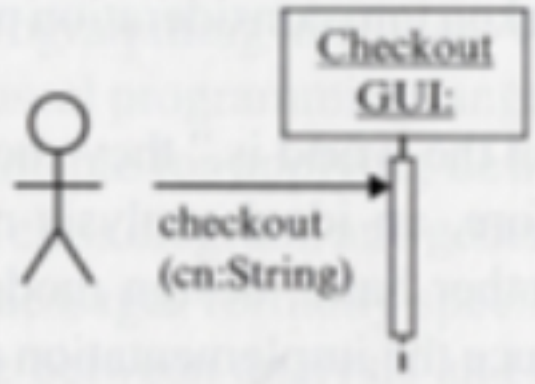
(b) Incorrect: the long rectangle beneath DBMgr should split into three as in (a).



(c) Incorrect: methods must be called to execute.



(d) Not preferred: the back dashed arrow line can be interpreted differently. Do as (a) is preferred.

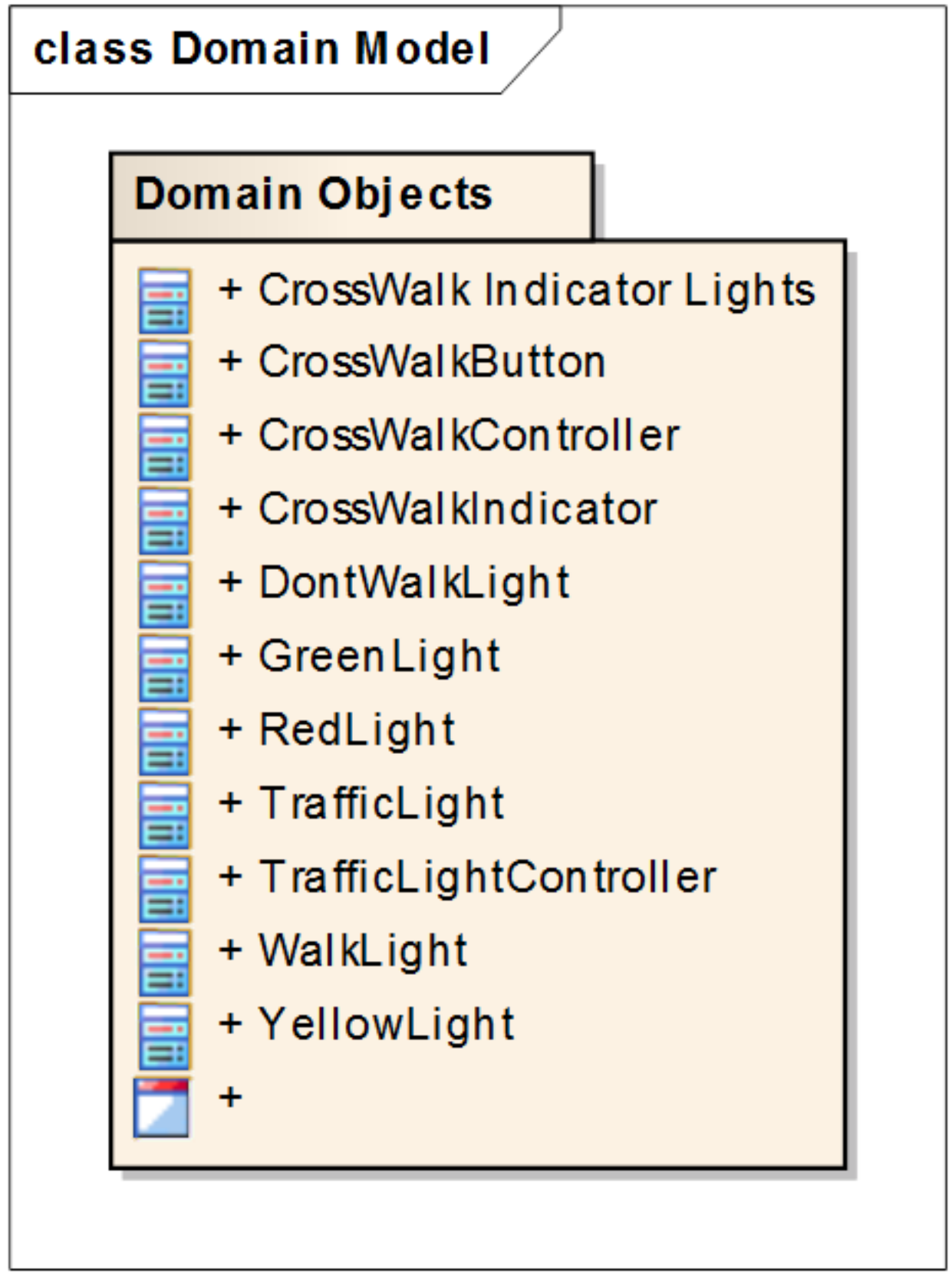


(e) Incorrect: an actor cannot call a function of an object; should use a dashed line and stereotype message.

FIGURE 9.6 Correct and incorrect uses of notations

Defining Operations

- Provide a set of orthogonal interface operations
- Hide the internal class structure
- All messages into an object must be accepted
  - Each message must have a corresponding acceptor operation
- Mutators and assessors should be provided as is necessary


class Domain Model
Domain Objects
+ CrossWalk Indicator Lights
+ CrossWalkButton
+ CrossWalkController
+ CrossWalkIndicator
+ DontWalkLight
+ GreenLight
+ RedLight
+ TrafficLight
+ TrafficLightController
+ WalkLight
+ YellowLight
+

A UML Design

class Class Model