Software Analysis and Design (CS:3004)

Timing Diagram

Timing Diagram:

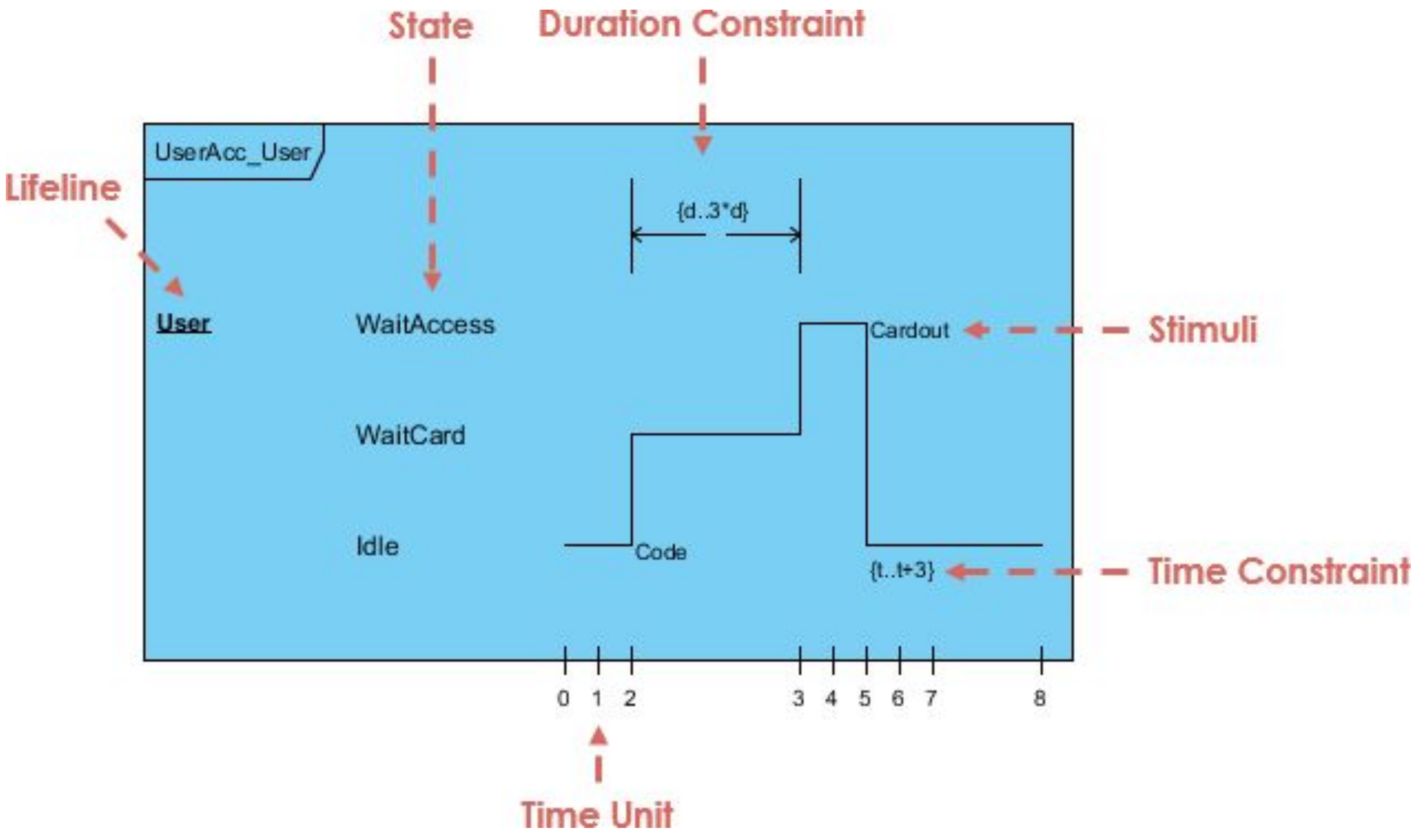
- A **timing diagram** in the Unified Modeling Language is a specific type of interaction diagram, where the focus is on timing constraints.
- It is used to explore the behaviors of objects throughout a given period of time.
- A timing diagram is a special form of a sequence diagram. The differences between timing diagram and sequence diagram are the axes are reversed so that the time increases from left to right and the lifelines are shown in separate compartments arranged vertically.
- The timing diagram is available since UML version 2.0 and includes elements such as message, lifeline, timeline, and object or role.

Timing Diagram:

- Timing diagrams are a type of interaction diagram that emphasize detailed timing specifications for messages. They are often used to model real-time systems such as satellite communication or hardware handshaking.
- They have specific notation to indicate how long a system has to process or respond to messages, and how external interruptions are factored into execution.
- Unlike sequence diagrams, timing diagrams are read left to right rather than top to bottom.

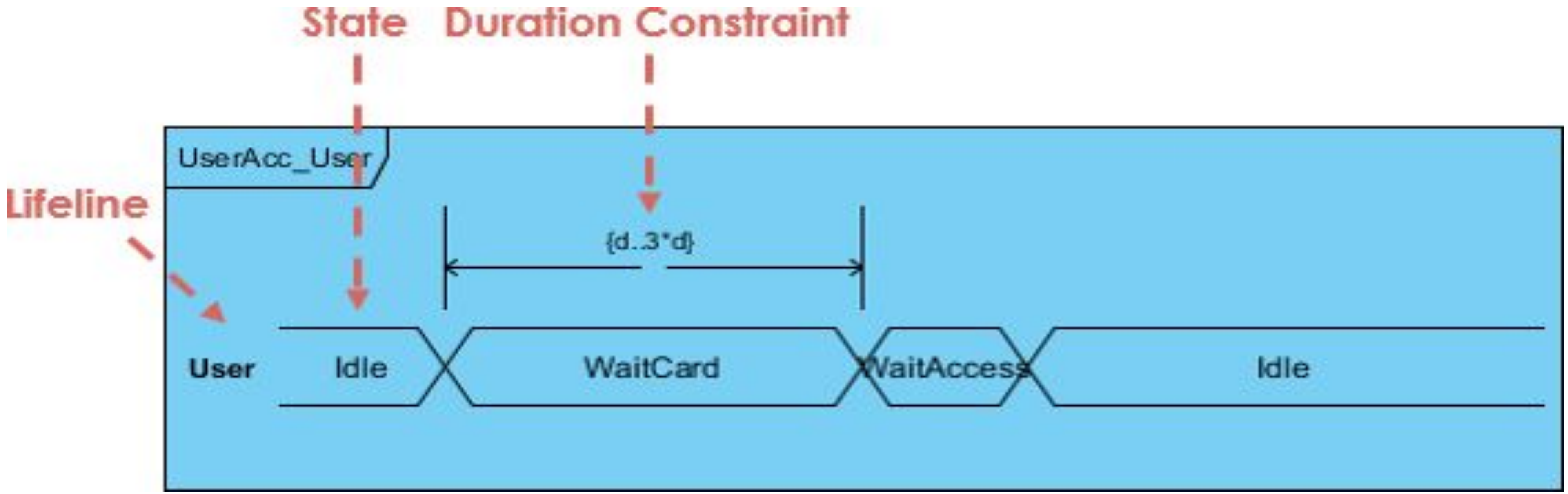
UML Timing Diagram: (Robust Notation)

robust: A complex line signal designed to show the transition from one state to another (can have many states).

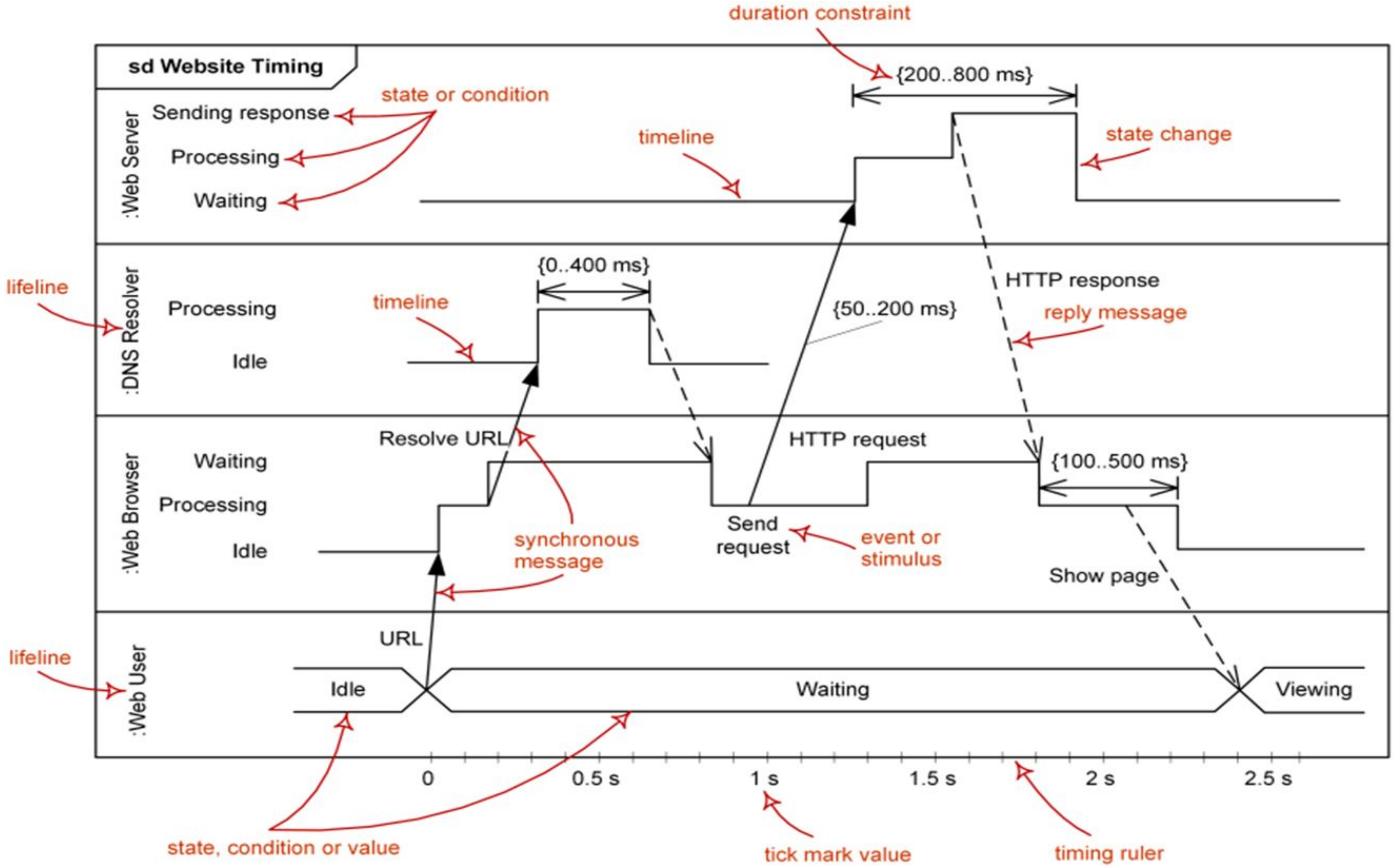


Alternative notation of UML Timing diagram: (Concise Notation)

Concise: A simplified signal designed to show the movement of data (great for messages).

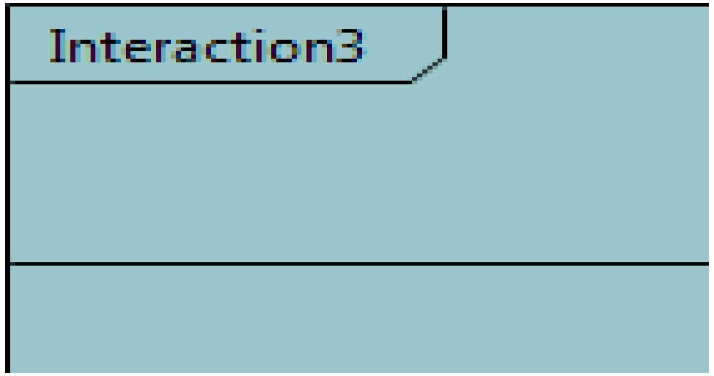


Major elements of timing UML diagram:



Frame/Interaction

The notation shows a rectangular frame around the diagram with a name in a compartment in the upper left corner



Lifeline: (full shape/ compact shape)

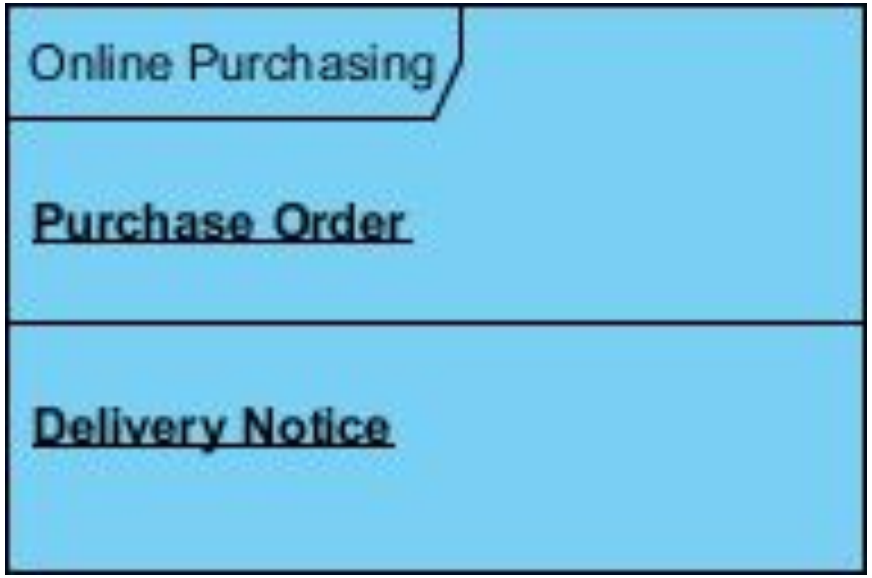
- There are two types of lifelines to show the lifeline of the components

- Lifeline is a named element which represents an individual participant in the interaction.

- Lifeline on the timing diagrams is represented by the name of classifier or the instance it represents. It could be placed inside diagram frame or a "swim lane".

Continue…

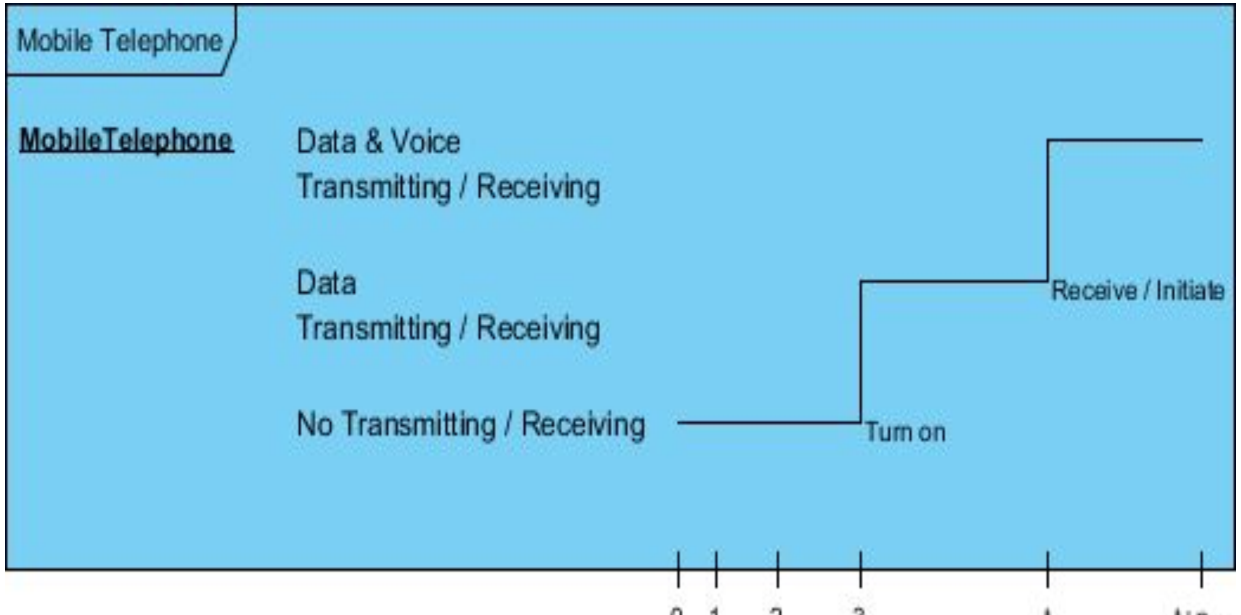
- Multiple lifelines may be stacked within the same frame to model the interaction between them.



Lifeline (full shape/ compact shape)

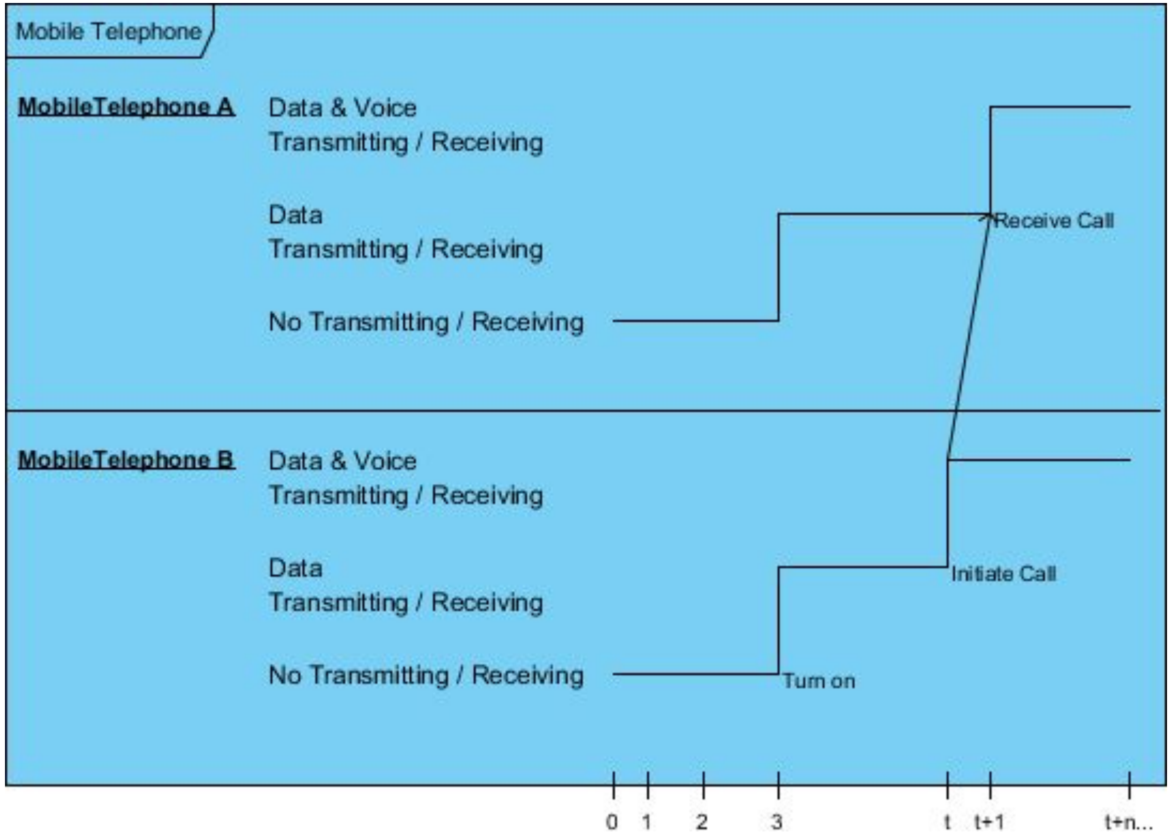
State Timeline:

A state or condition timeline represents the set of valid states and time. The states are stacked on the left margin of the lifeline from top to bottom.



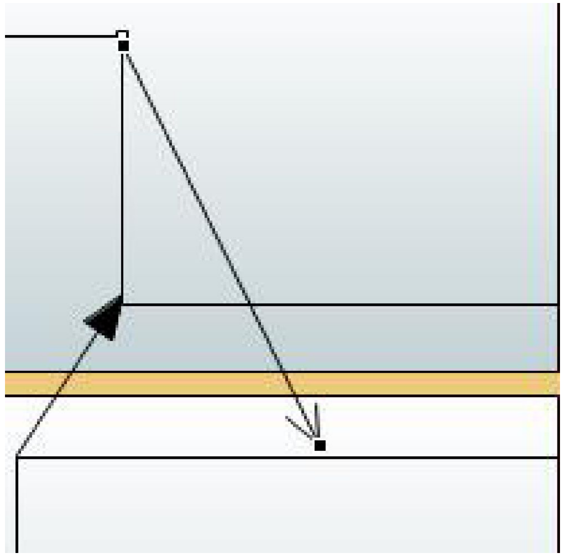
Multiple Compartments:

It is possible to stack several lifelines of different objects in the same timing diagram. One lifeline above the other. Messages sent from one object to another can be depicted using simple arrows. The start and the end points of each arrow indicate when each message was sent and when it was received.



Synchronous Message/ Reply Message

- Messages show interaction among diff lifelines. Message is represented as filled arrow and reply as unfilled arrow

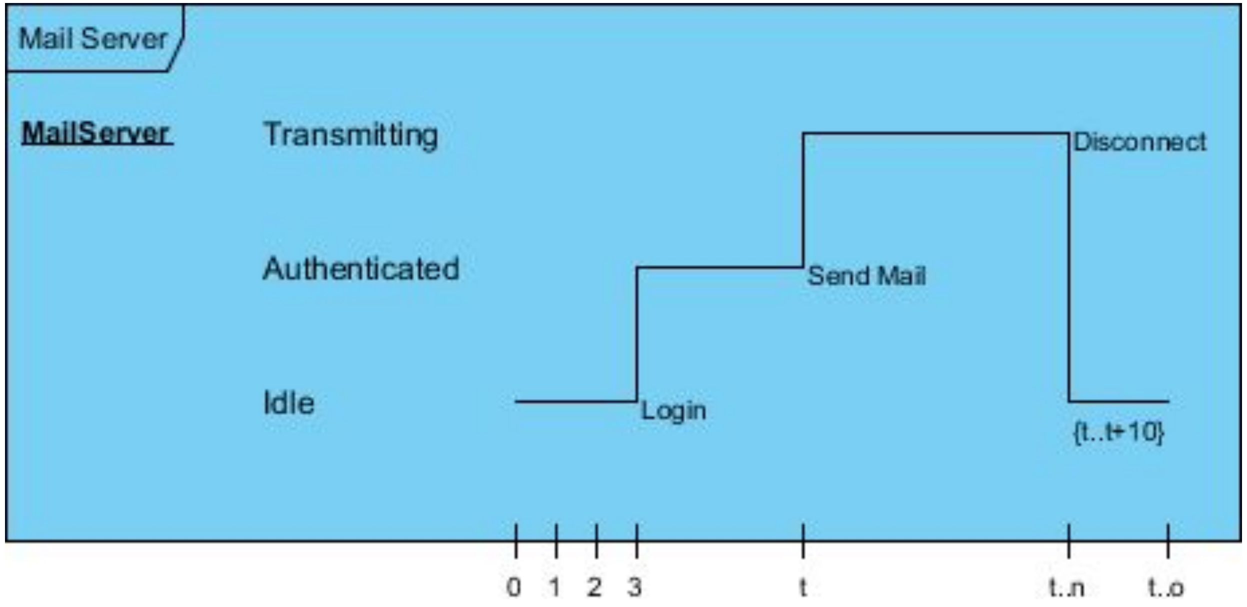


Timeline and Constraints:

- We can use the length of a timeline to indicate how long the object remains in a particular state by reading it from left to right. To associate time measurements, you show tick marks online the bottom part of the frame.

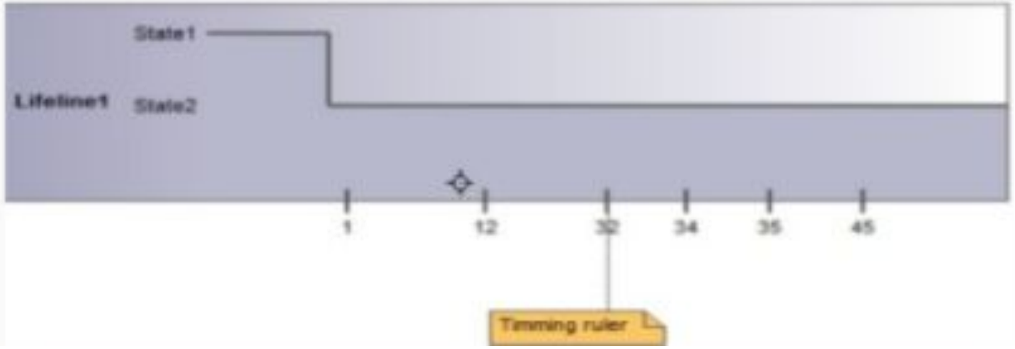
Continue…

The example below shows that the Login event is received three-time units after the start of the sequence. To show relative times, you can mark a specific instance in time using a variable name. The figure marks the time the sendMail event is received as time

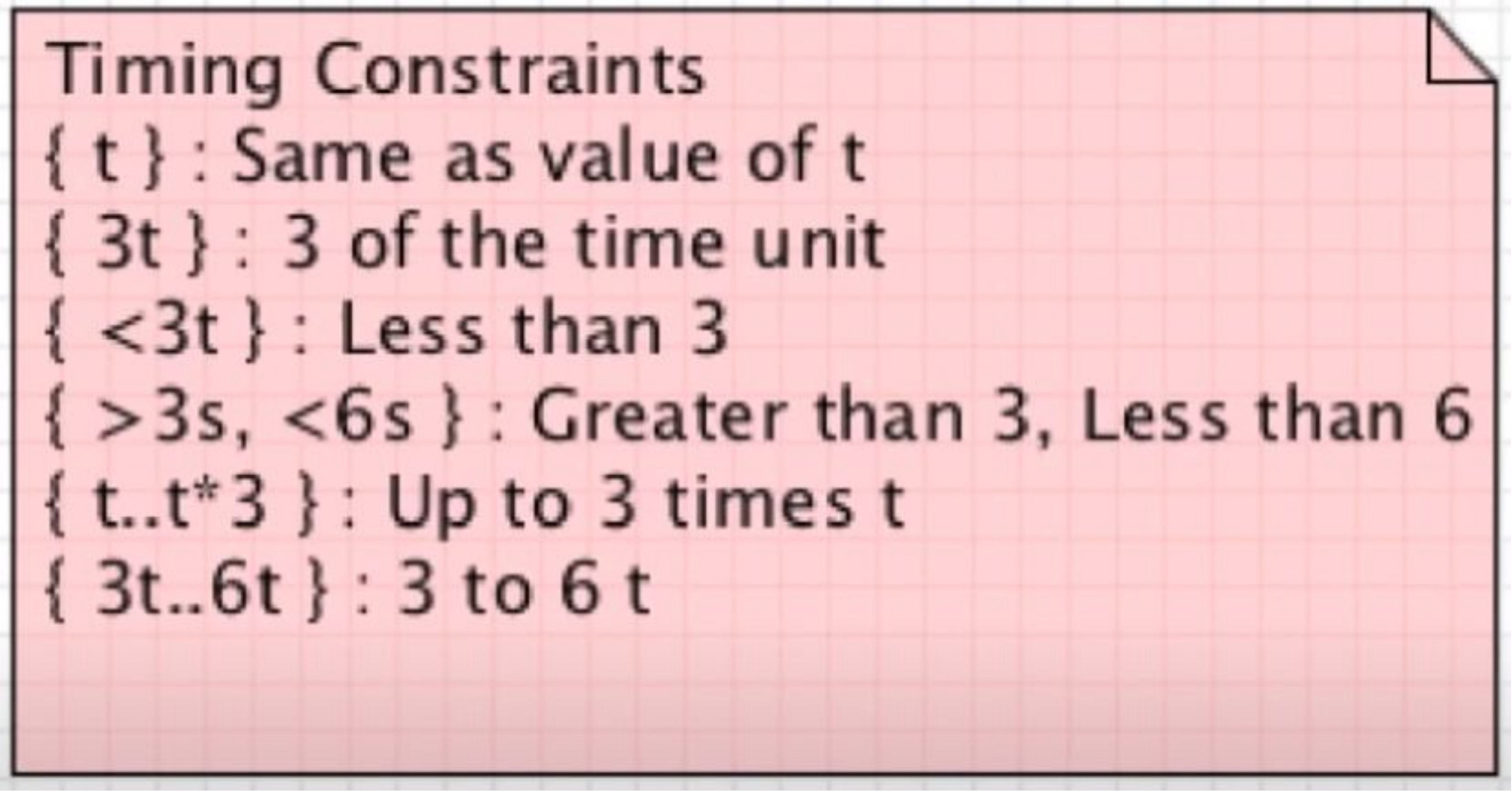


Timing Ruler:

A timing ruler is a graduated device for indicating the passage of time on a timing diagram. The graduations are called ticks and are typically placed on the lower edge of the diagram frame. Time runs from left to right. The scale and time units (ticks) depend upon what the diagram represents and may be sub-second units such as milliseconds or larger time divisions such as hours, days, months or even years.

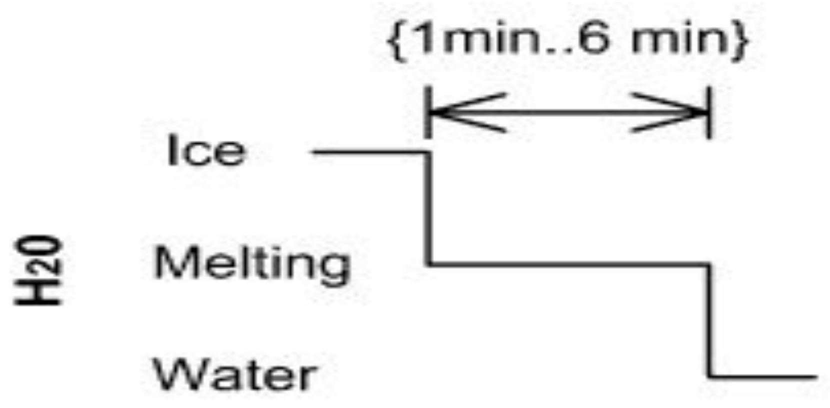


Timing constraints



Duration Constraint:

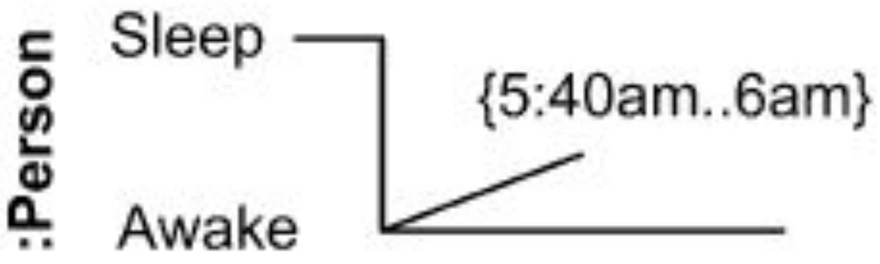
- Duration constraint is an interval constraint that refers to a duration interval. The duration interval is duration used to determine whether the constraint is satisfied.



Ice should melt into water in 1 to 6 minutes

Time Constraint

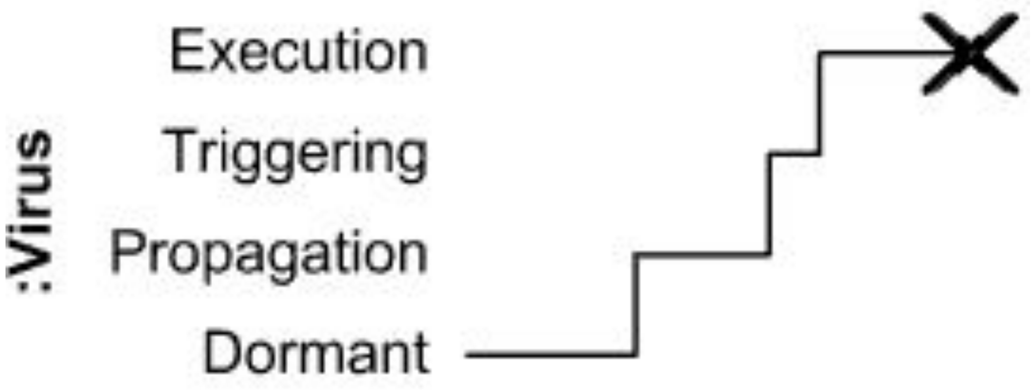
- Time constraint is an interval constraint that refers to a time interval. The time interval is time expression used to determine whether the constraint is satisfied.



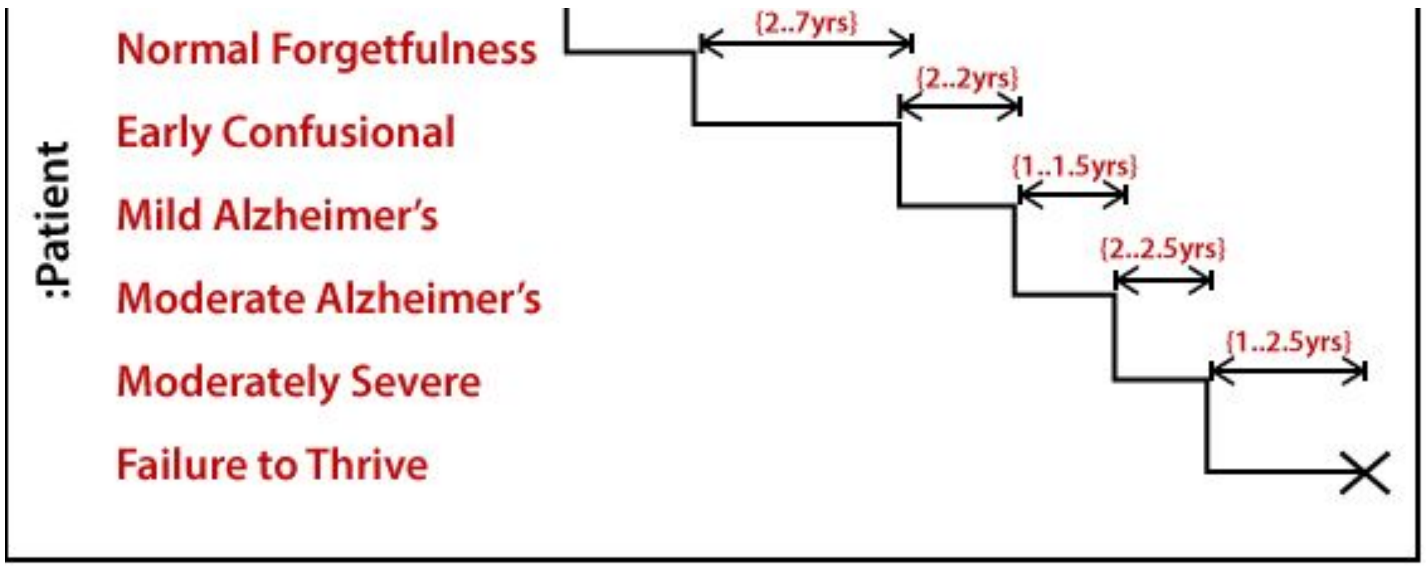
Person should wake up between 5:40 am and 6 am

Destruction Occurrence:

- Destruction occurrence is a message occurrence which represents the destruction of the instance described by the lifeline. It may result in the subsequent destruction of other objects that this object owns by composition. No other occurrence may appear after the destruction event on a given lifeline.
- Notation: The destruction event is depicted by a cross in the form of an X at the end of a timeline.



A timing diagram example of a medical domain that depicts different stages of Alzheimer's disease (AD) is explained below.




LET'S
DRAW

1. Find the states

- 1. Customer Inserts Card
- 1A. Card is Invalid
- 1B. Eject Card
- 2. Card is Validated
- 3. Customer Enters PIN
- 3A. PIN is Invalid
- 3B. PIN is Invalid 3 Times
- 3C. Card Marked as Stolen
- 4. PIN is Validated
- 5. Account is Selected
- 6. Amount is Selected
- 6A. Over Daily Maximum
- 6B. Over Account Funds Available
- 6C. Over Funds in Machine
- 6D. Ask for New Amount
- 7. Provide $10 or $20
- 8. Provide Funds
- Provide Receipt
- 10. Eject Card

2. Find the participants

- Customer
- ATM/Card reader
- Server/BankAccountDB

3. Match the states to participants

- Customer
  - Card in Hand
  - Insert Card
  - Enter PIN
  - Select Account
  - Select Amount

3. Match the states to participants

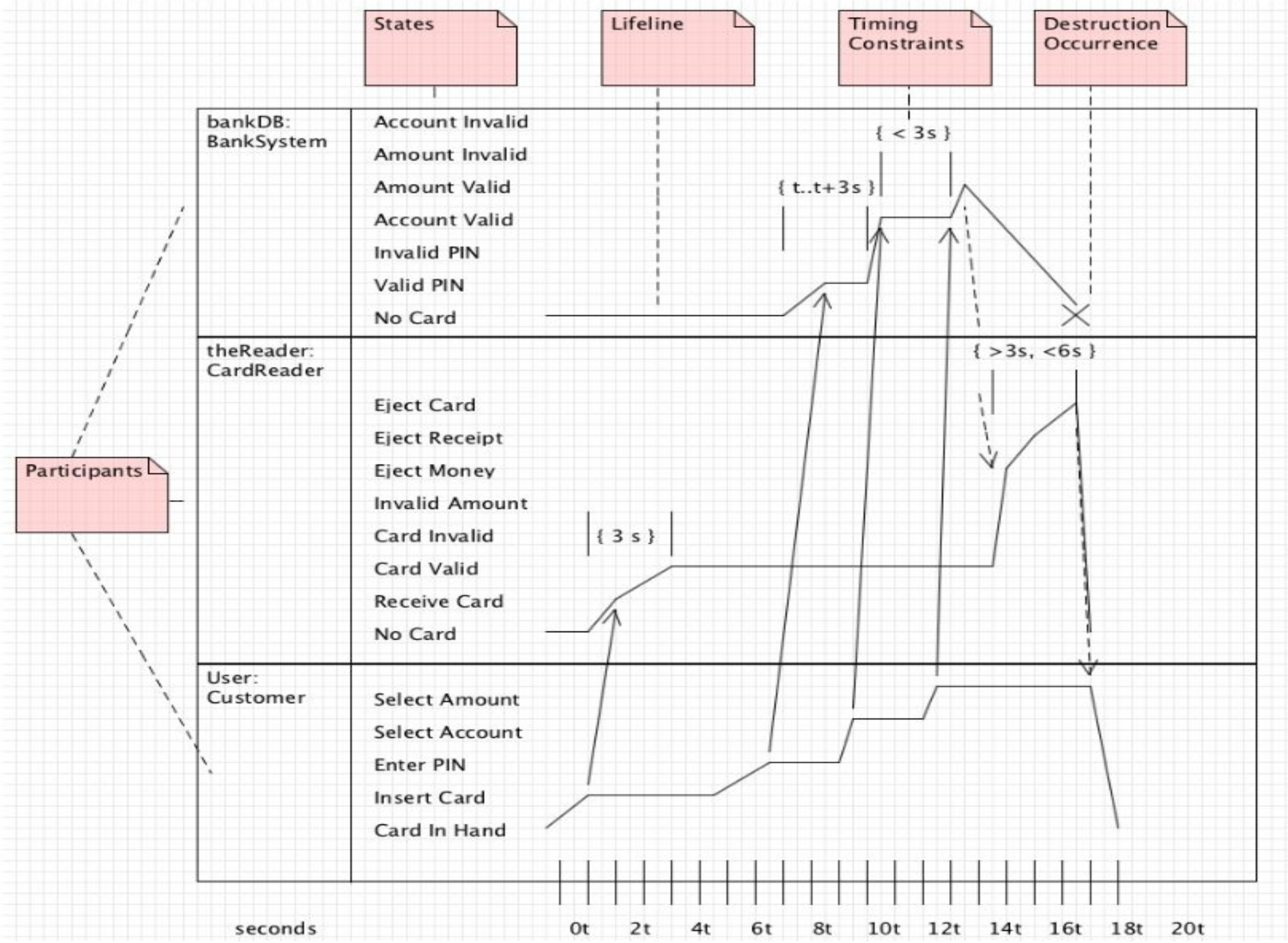
- Card Reader
  - No Card
  - Receive Card
  - Card Valid
  - Card Invalid
  - Invalid Amount
- Eject Money
- Eject Receipt
- Eject Card

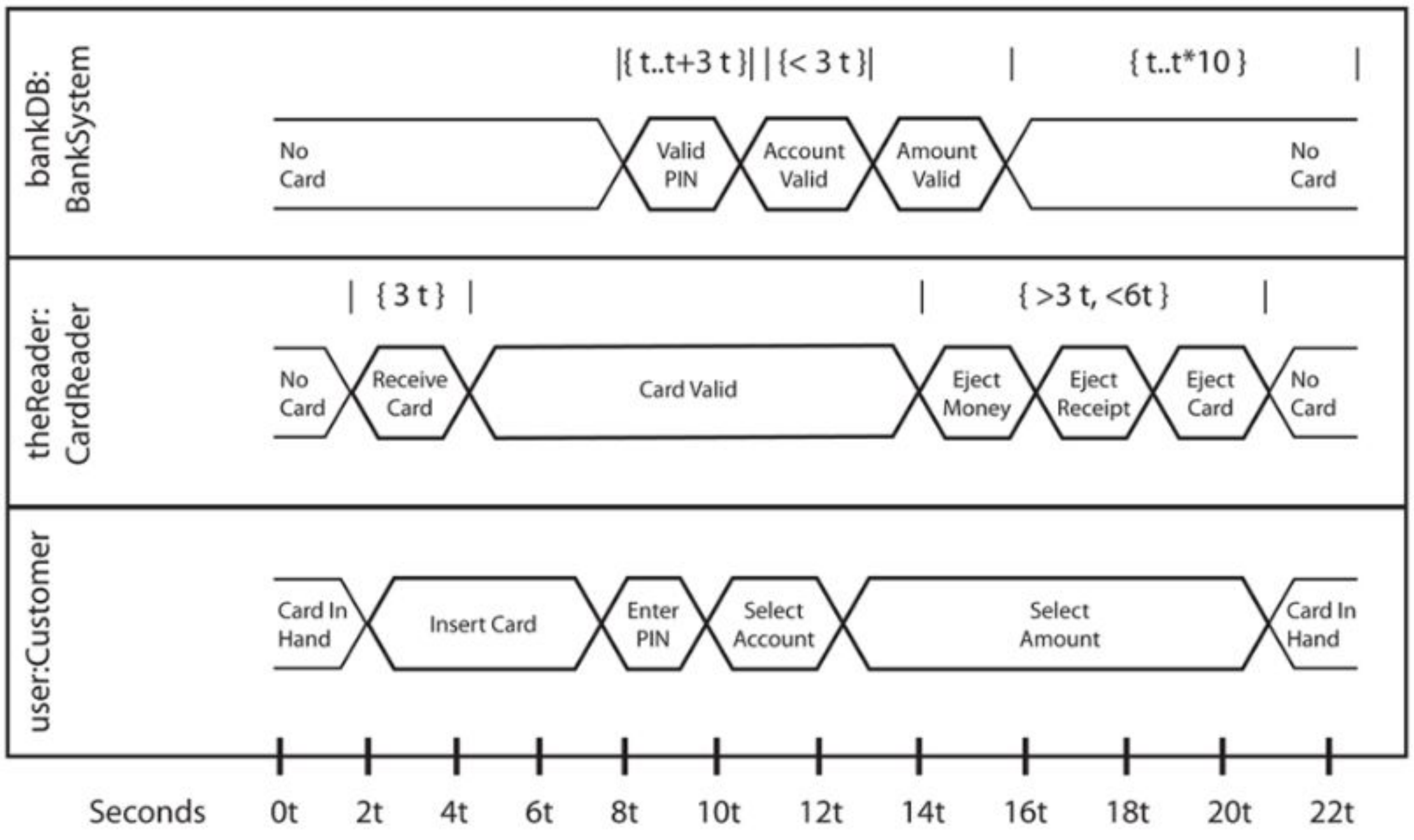
3. Match the states to participants

- Bank System
  - No Card
  - Valid PIN
  - Invalid PIN
  - Account Valid
  - Amount Valid
  - Amount Invalid
  - Account Invalid

Steps of Execution

1. Customer Inserts Card
1A. Card is Invalid
1B. Eject Card
2. Card is Validated
3. Customer Enters PIN
3A. PIN is Invalid
4. PIN is Validated
5. Account is Selected
5A. Account is Valid
5B. Account is Invalid
6. Amount is Selected
6A. Amount is Valid
6B. Amount is Invalid
7. Eject Money
8. Eject Receipt
9. Eject Card


States
Lifeline
Timing Constraints
Destruction Occurrence
Participants
bankDB: BankSystem
Account Invalid
Amount Invalid
Amount Valid
Account Valid
Invalid PIN
Valid PIN
No Card
{ < 3s }
{ t..t+3s }
theReader: CardReader
{ >3s, <6s }
Eject Card
Eject Receipt
Eject Money
Invalid Amount
Card Invalid
Card Valid
Receive Card
No Card
{ 3 s }
User: Customer
Select Amount
Select Account
Enter PIN
Insert Card
Card In Hand
seconds
0t
2t
4t
6t
8t
10t
12t
14t
16t
18t
20t


bankDB: BankSystem
|{ t..t+3 t }| |{< 3 t }|
| { t..t*10 } |
No Card
Valid PIN
Account Valid
Amount Valid
No Card
theReader: CardReader
| { 3 t } |
| { >3 t, <6t } |
No Card
Receive Card
Card Valid
Eject Money
Eject Receipt
Eject Card
No Card
user:Customer
Card In Hand
Insert Card
Enter PIN
Select Account
Select Amount
Card In Hand
Seconds
0t
2t
4t
6t
8t
10t
12t
14t
16t
18t
20t
22t

Scenario:

Part of the lift specification used in this scenario. The requirements maybe described “...A part of the lift system contains two objects: the floor sensor and the lift. The lift movement states are separated into three steps: moving up(2), stop at floor and moving down(1). The floor sensor has two states: on and off. The relation between the lift movement and the floor sensors means a user presses a button to request the lift from floor 1, the lift starts moving down from the current floor; lift must be arrive at requested floor within between 2 – 5 seconds after the lift starts moving, the floor sensor of the current floor will turn off within given time frame .”