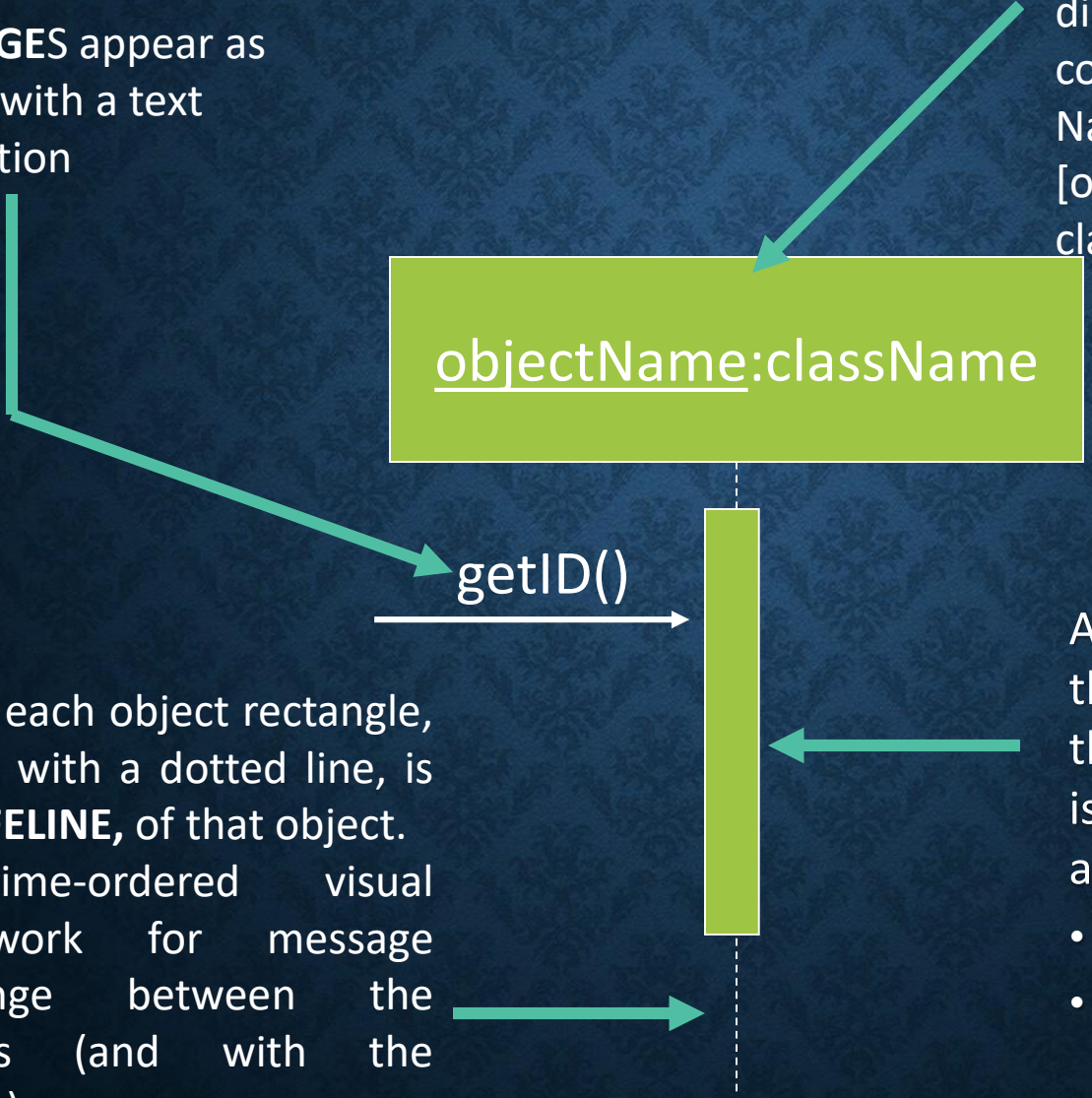
SEQUENCE DIAGRAM / INTERACTION DIAGRAM

SEQUENCE DIAGRAM

- Captures dynamic behavior (time-oriented)
- Purpose
  - Model flow of control
  - Illustrate typical scenarios
- A sequence diagram shows
  - an interaction arranged in time sequence,
  - the objects (not classes)
  - and the messages that pass between them when an interaction occurs

OBJECTS are shown in rectangles on the top of the diagrams, Each rectangle contains
Name (always underlined)
[objects are underlined not class]

MESSAGES appear as arrows with a text description



Below each object rectangle, shown with a dotted line, is the **LIFELINE,** of that object.
A time-ordered visual framework for message exchange between the objects (and with the system)

A narrow vertical line called the **ACTIVATION** represents the period of time an object is actually performing an action

- Directly
- Or through an intermediary (such as another object)

MESSAGE TYPES

1. Simple Message

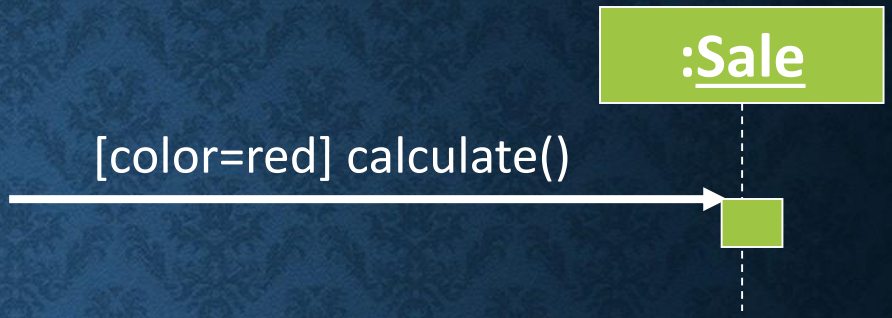
Control is passed from one object to another without providing details

2. Self Message

A message being sent from an object to itself

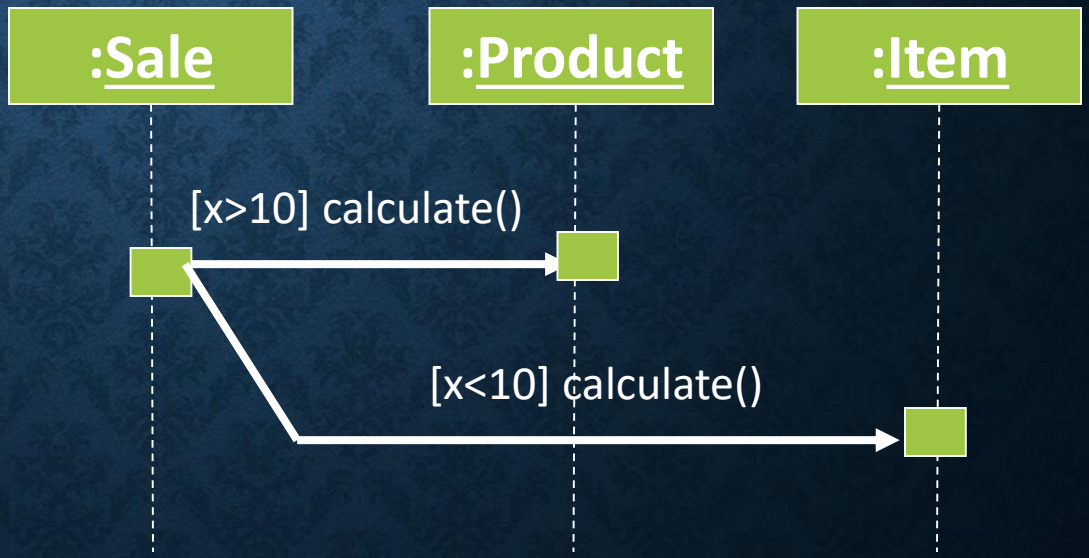
3. Conditional Message

A condition can be expressed with this message



4. Mutually exclusive conditional messages

Notation for this kind of message is an angled line emerging from a common point

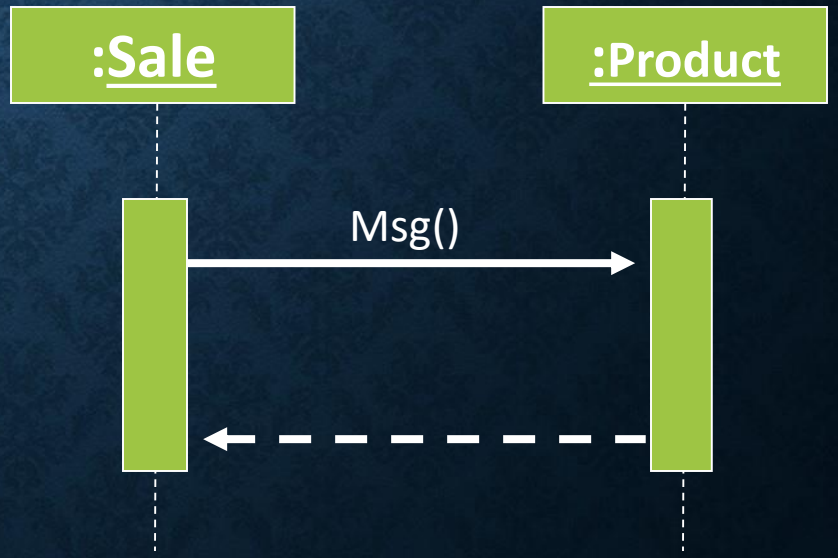


5. Object destruction message

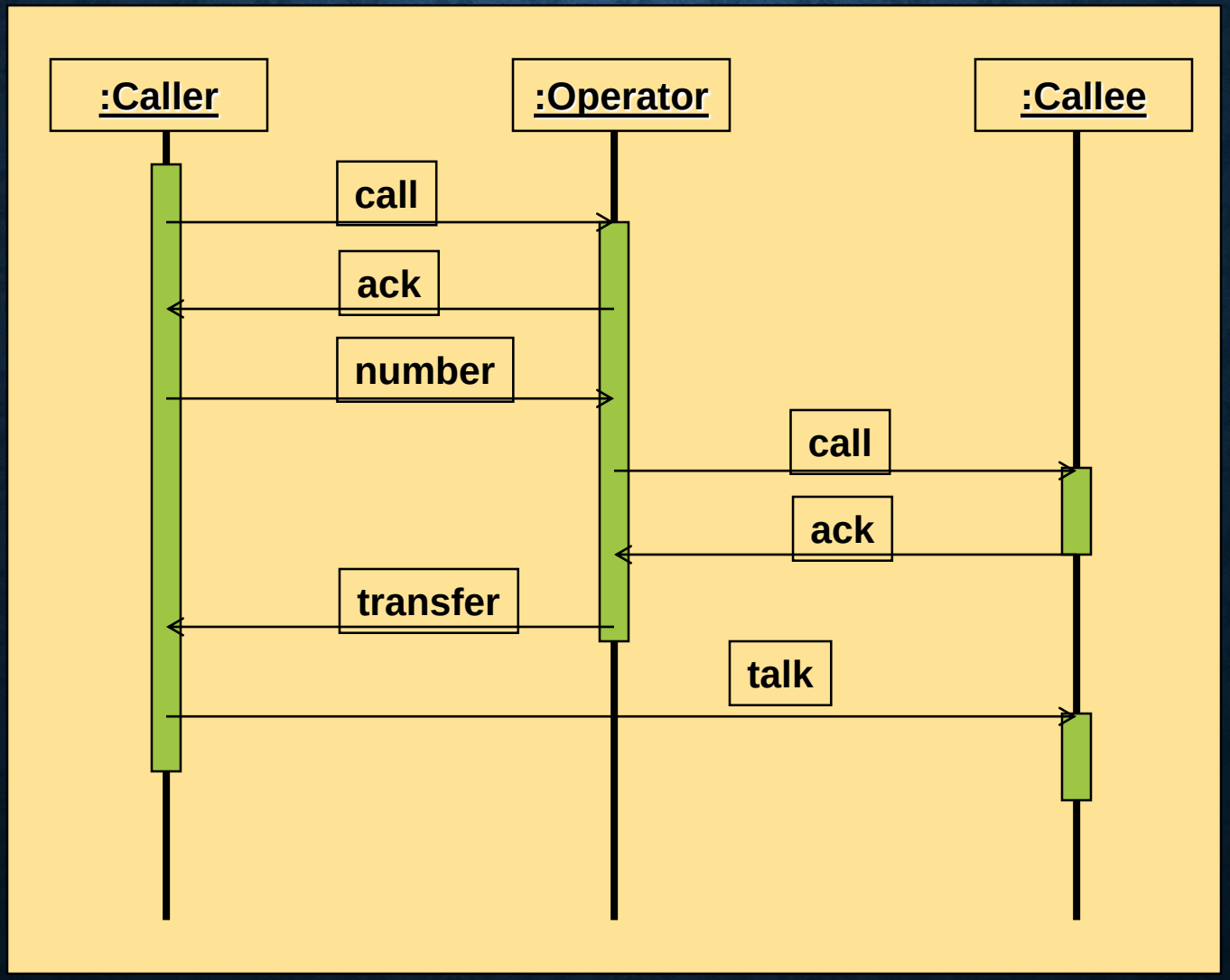
A message with <<destroy>> stereotype and a short lifeline indicates an explicit object destruction

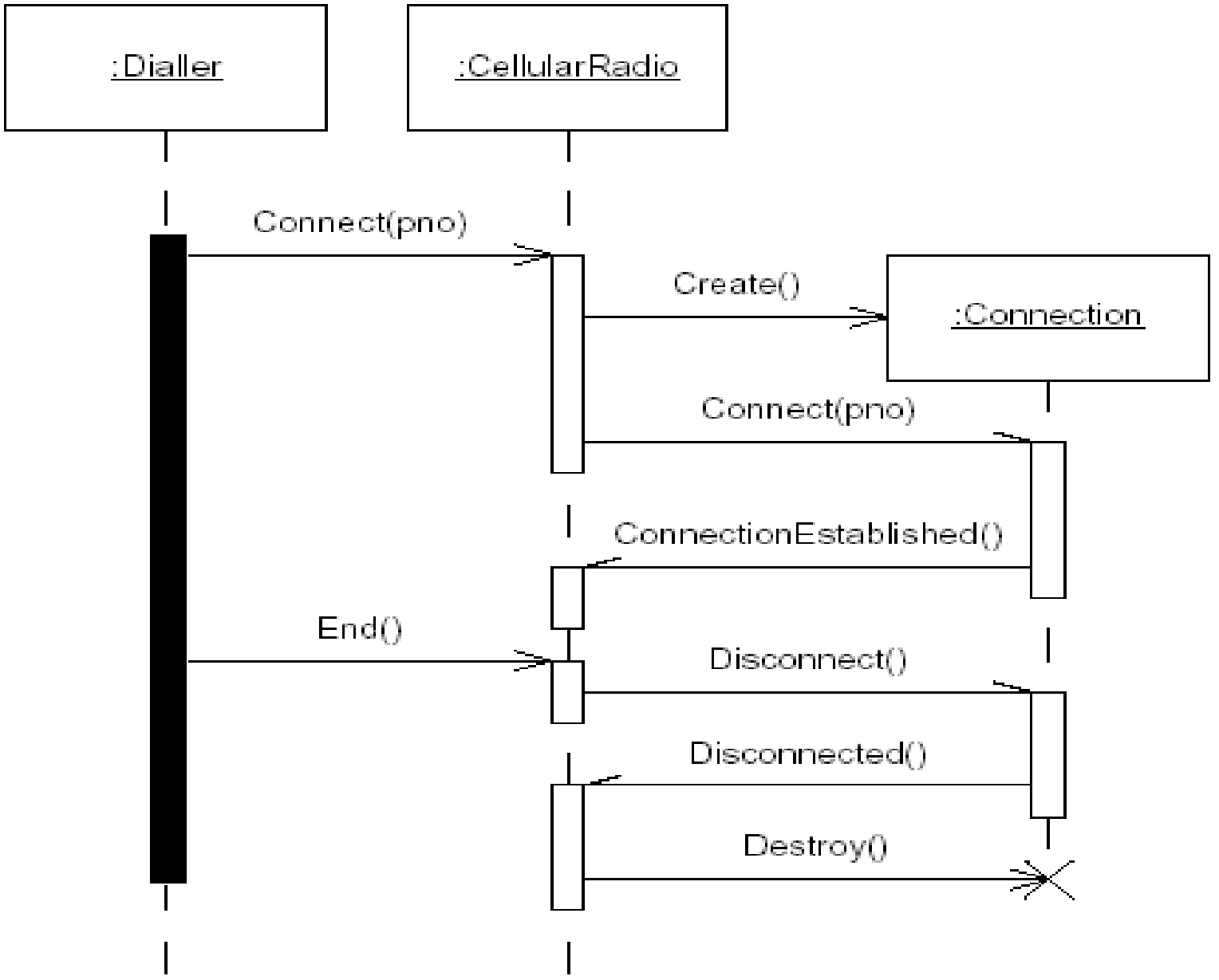
6. Return Message

This message indicates a return from a procedure call



SEQUENCE DIAGRAM: EXAMPLE



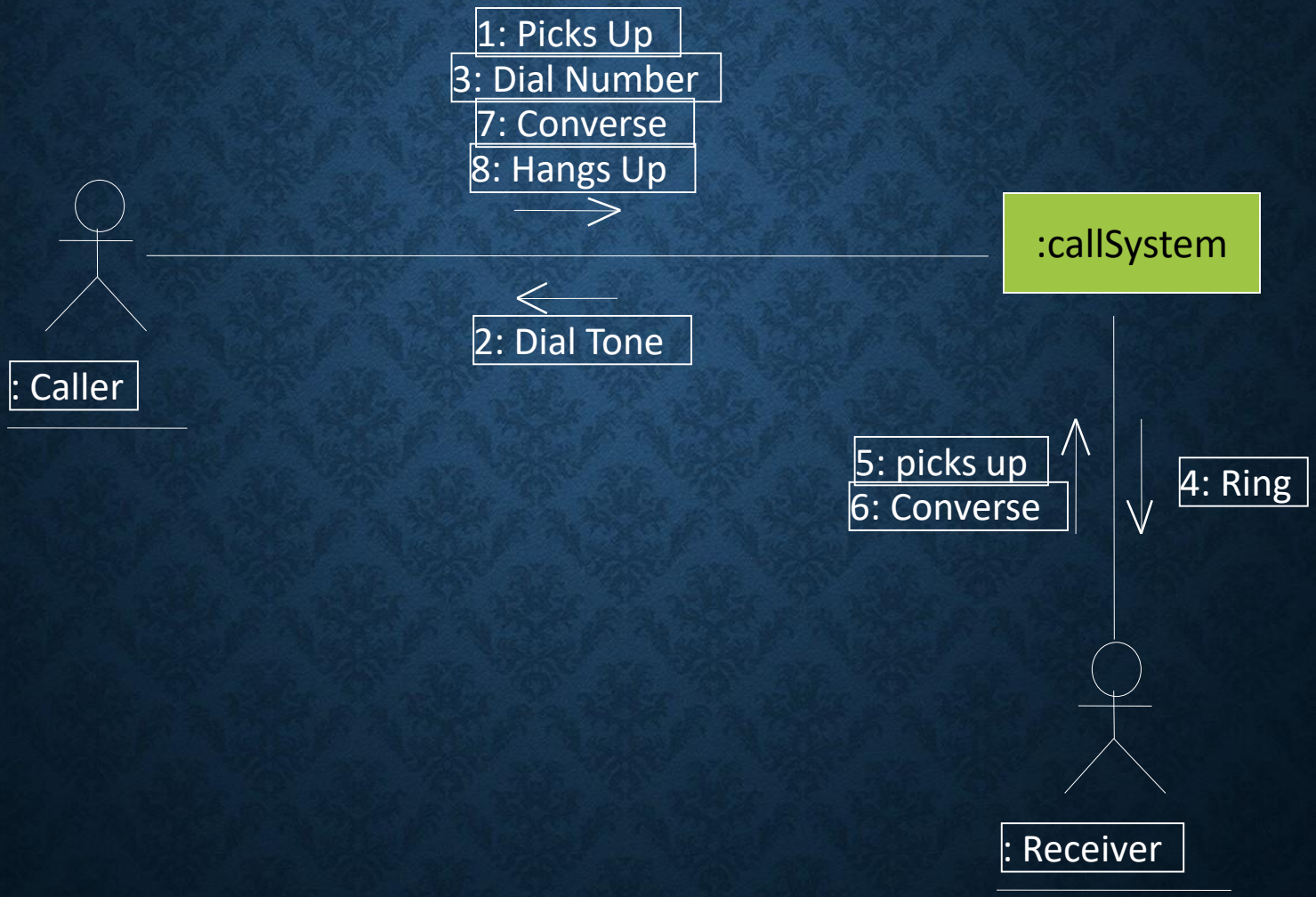

:Dialler
:CellularRadio
Connect(pno)
Create()
:Connection
Connect(pno)
ConnectionEstablished()
End()
Disconnect()
Disconnected()
Destroy()

COLLABORATION DIAGRAM

A Collaboration Diagram also shows the passing of messages between objects BUT **focuses on the order** of objects and their messages instead of **the time sequence**

- Basic Notations
  - Objects
  - Links
  - Messages
- Basic Steps
  - Identify roles involved
  - Identify interactions & concurrent threads
  - Sequentially number them and mark with arrows for directions

COLLABORATION DIAGRAM



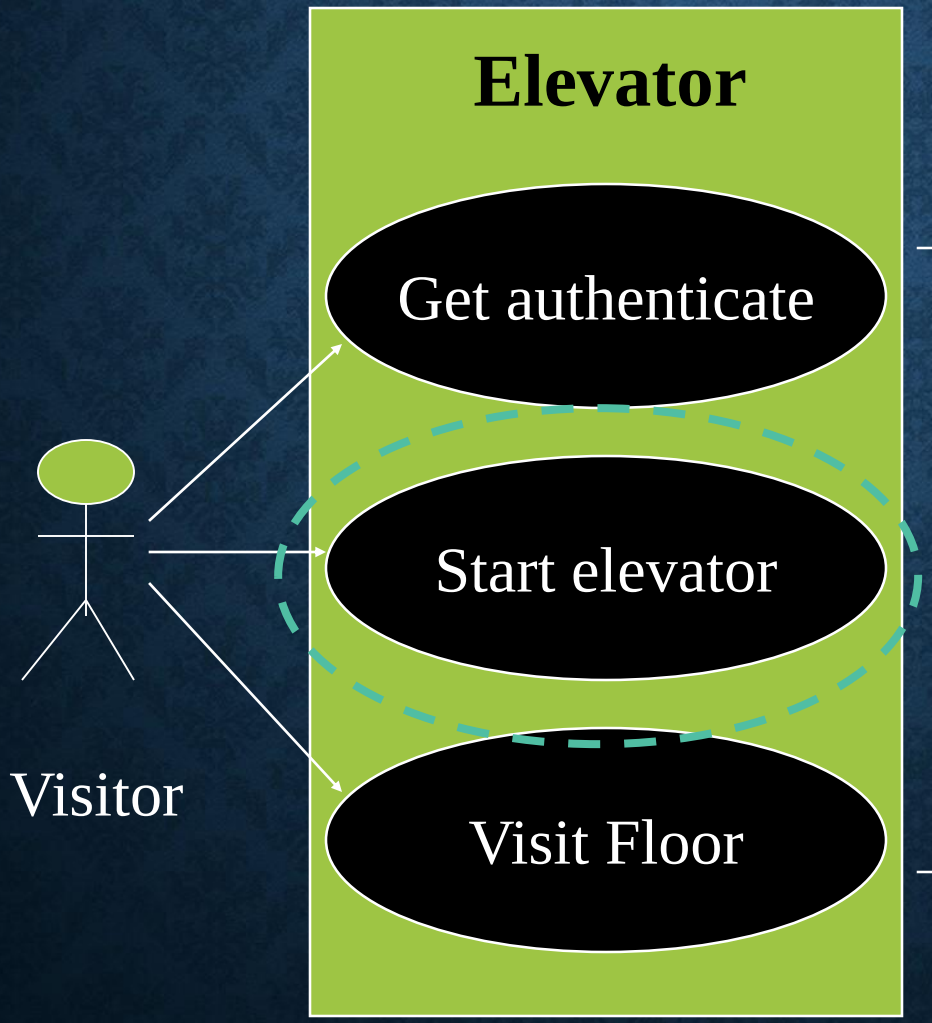
ELEVATOR PROBLEM

PROBLEM STATEMENT

A product is to be installed to control n elevators in a building within m floors. The problem concerns the logic required to move elevators between floors according to the following constraints

1. Each floor , except the first floor and the top floor, has two buttons, one to request an up elevator and one to request a down elevator. These illuminates when pressed. The illumination is cancelled when an elevator visits the floor and then moves in the desired direction
2. Each elevator has a set of m buttons, one for each floor. These illuminates when pressed and cause the elevator to visit the corresponding floor. The illumination is cancelled when corresponding floor is visited by the elevator
3. There is a display window that shows the current floor being visited by the elevator
4. When an elevator has no request, it remains at its current floor with its door closed

IDENTIFY ACTOR, GOALS, USE CASES



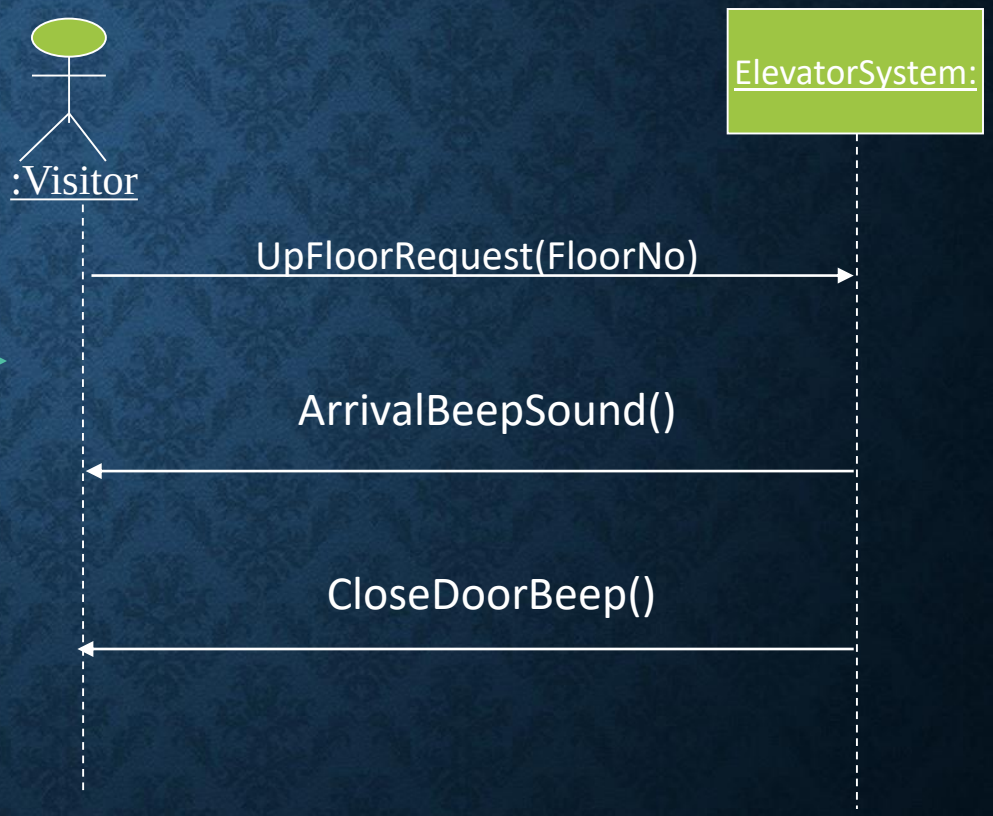
Goals

- He get authentication and pass the security system
- He visits the elevator and wants to move to the destination floor without interruption

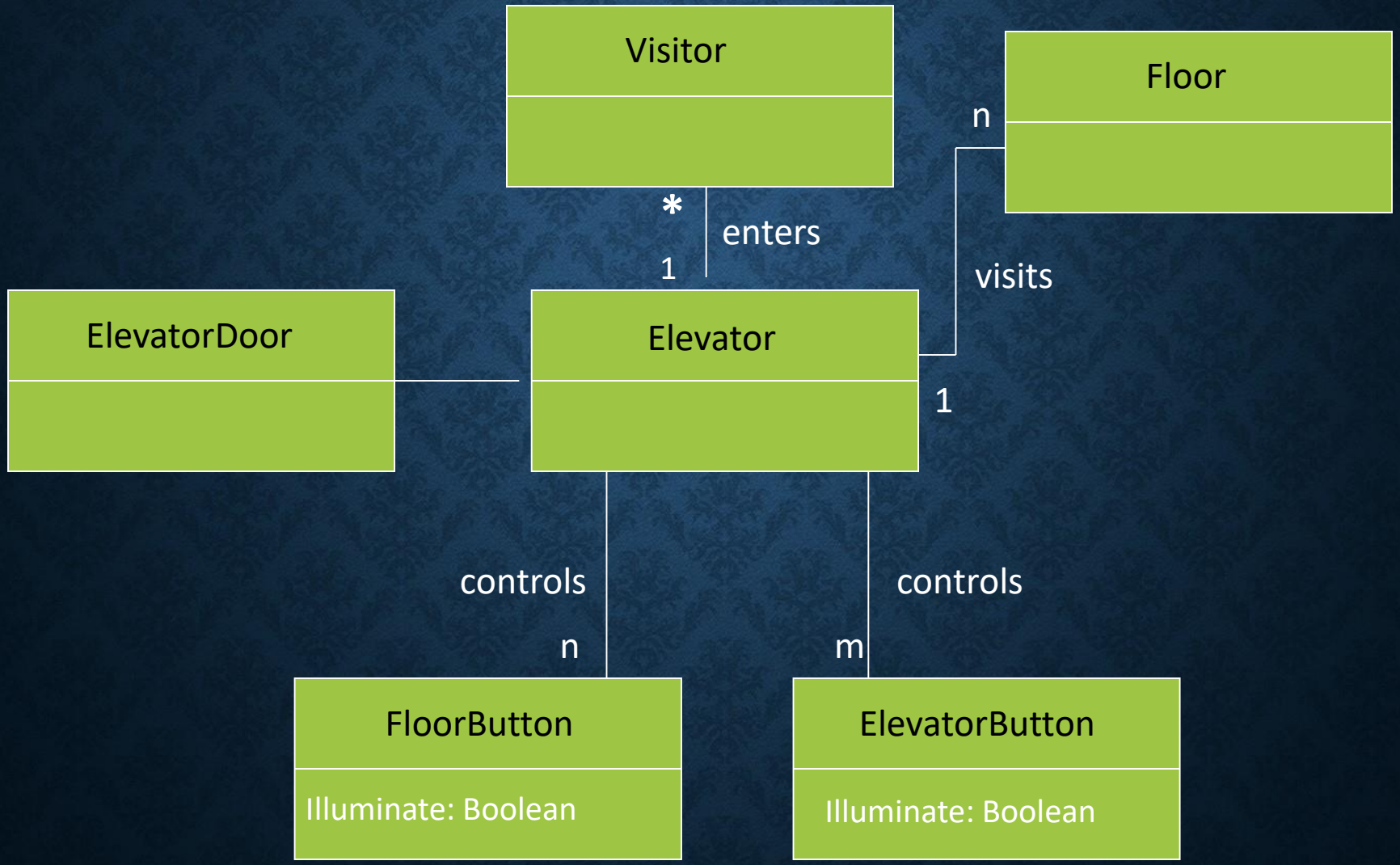
<<UC-1>>START ELEVATOR SSD

Scenario

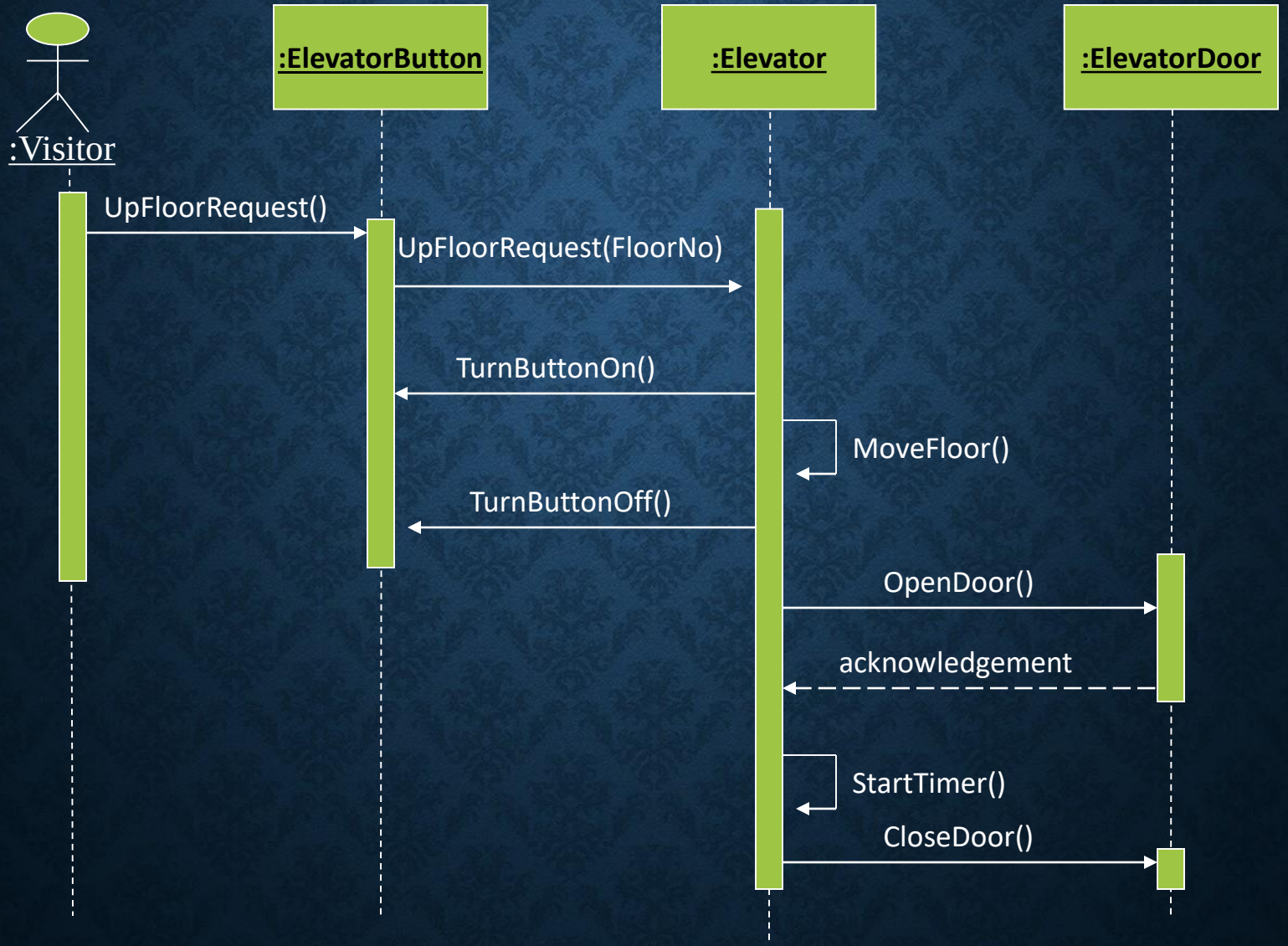
- Visitor A presses the Up floor button at floor 3 to request an elevator, he wishes to go to floor 7
- The up floor button turned on
- An elevator arrives at floor 3
- The up floor button turn off
- The elevator doors open
- The timer starts
- Visitor A enters the elevator
- The elevators door close after the time out



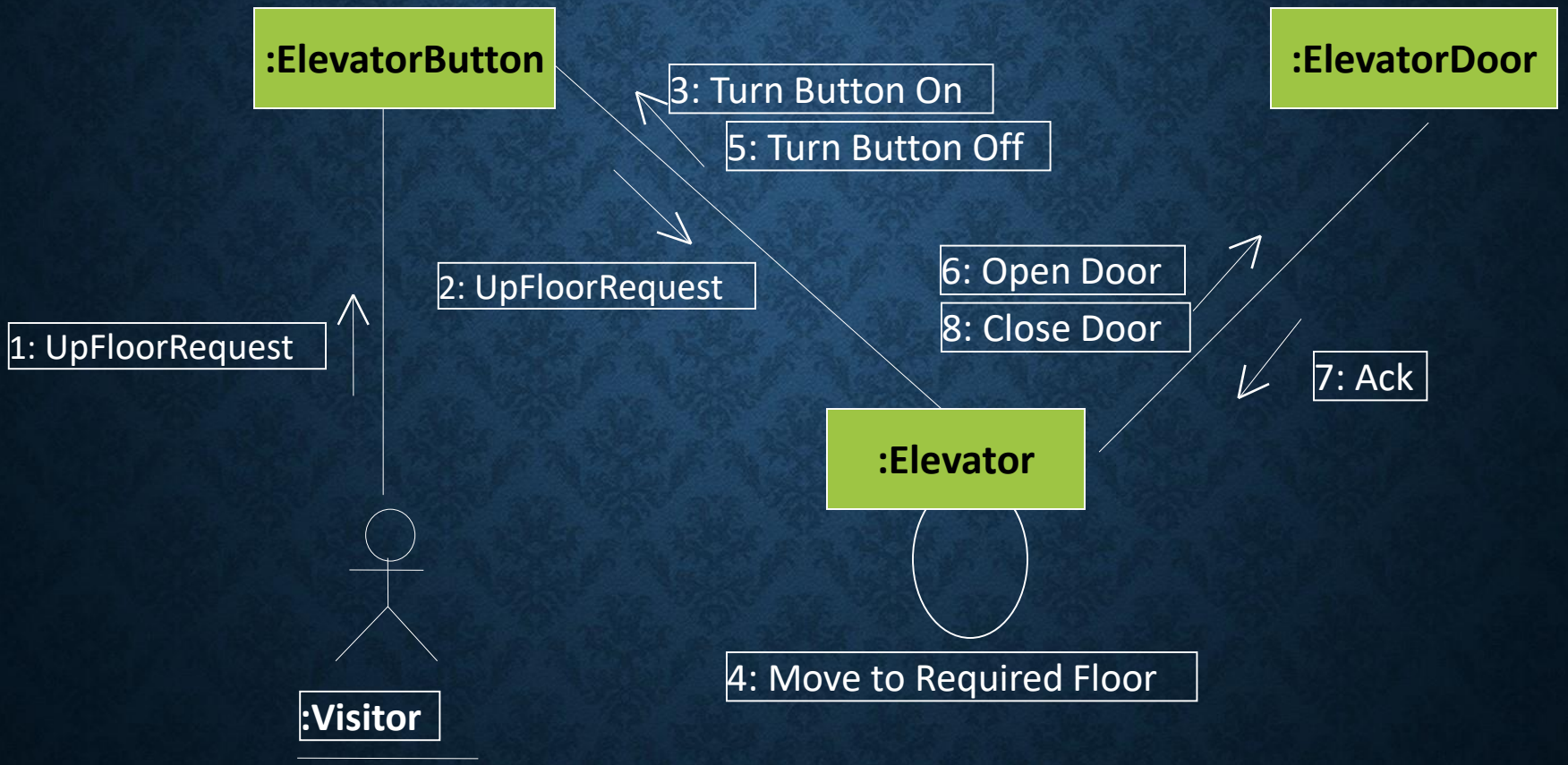
PARTIAL DOMAIN MODEL: ELEVATOR PROBLEM



<<UC-1>>START ELEVATOR SEQUENCE DIAGRAM



<<UC-1>>Start Elevator Collaboration Diagram



REFERENCES

- Applying UML and Patterns by Craig Larman
  - Chapter 13: 13.1, 13.2, 13,9
  - Chapter 15: 15.1, 15.5, 15.6, 15.7
- Other Reference
  - Object-Oriented and Classical Software Engineering by Stephen. R. Schach
- Site Reference
  - http://www.agilemodeling.com/artifacts/sequenceDiagram.htm