unit2: chapter2
Interaction Model

- Interactions can be modeled at different levels of abstraction
  - Use case
  - Sequence
  - Activity

Use case

❖Use cases describe how a system interacts with outside actors

❖Each use case represents a piece of functionality that a system provides to its users

Sequence diagram

- Sequence diagram show the messages exchanged among a set of objects over time
- Also it shows behavior sequences seen by users of a system

Activity diagram

- It shows the flow of control among the steps of a computation
- Activity diagrams can show data flows as well as control flows

- Use case model

Actors
Use cases
Use case diagram

Actors

- An actor is a direct external user of a system
- Actor can be person, devices, and other system
- Example :

  Juice vending machine

Use Cases

- Use case is a coherent piece of functionality that a system can provide by interacting with actors
- Example:

  Customer: buy a beverage

  customer: insert money

  customer: makes beverage selection

  repair technician: scheduled maintenance

Use cases

- Each use case involves one or more actors
- Example :

  buy beverage: customer

  perform scheduled maintenance: repair technician

  make a call: caller

  receiver

Actors

- The actors need not all be persons
- Example:

  Use case- make a trade: customer and stock exchange

- A use case involves a sequence of messages among the system and its actors
- Example:

use case- buy beverage:

- ❑customer inserts coin
- ❑vending machine displays the amount deposited
- ❑Customer select the beverage
- ❑Vending machine dispenses it
- ❑Issues change if necessary

- Error conditions are also part of a use case
- Example:

  If customer selects the beverage whose supply is exhausted

  Vending machine displays warning message
- Vending transaction can be cancelled
- Example:

  Customer can push the coin return on vending machine at any time before a selection has been accepted

- Use case brings together all of the behavior relevant to a system functionality
    - Normal behavior
    - Exception conditions
    - Error conditions
    - Cancellation of request

- **Buy a beverage:** the vending machine delivers a beverageAfter a customer selects and pays for it
- **Perform scheduled maintenance:** A repair technician performs the periodic service on the vending machine necessary to keep It in good working condition
- **Make repairs:** A repair technician performs the unexpected service on the vending Machine Necessary to repair a problem in its operation
- **Load items:** A stock clerk adds items into the vending machine to replenish its stock of beverages

Use Case: Buy a beverage

Summary: The vending machine delivers a beverage after a customer selects and pays for it

Actors: Customer

Preconditions: The machine is waiting for money to be inserted

Description: The machine starts in the waiting state in which it displays the message " Enter coins" A customer inserts coins into the machine. The machine displays the total value of money entered and lights up the buttons for the items that can be purchased for the money inserted. The customer pushes a button. The machine dispenses the corresponding item and make changes, if the cost of the item is less than the money inserted.

Exceptions:

Canceled: If the customer presses the cancel button before an item has been selected, the customer's money is returned and the machine resets to the waiting state.

Out of stock: if the customer presses a button for an item that costs more than the money inserted, the message "You must insert $nn.nn more for that item" is displyed, where nn.nn is the amount of additional money needed. The machine continues to accept coins or a selection

No change: if the customer has inserted enough money to buy the item but the machine cannot make the correct change, the message "Cannot make correct change" is displayed and the machine continues to accept coins or a selection.

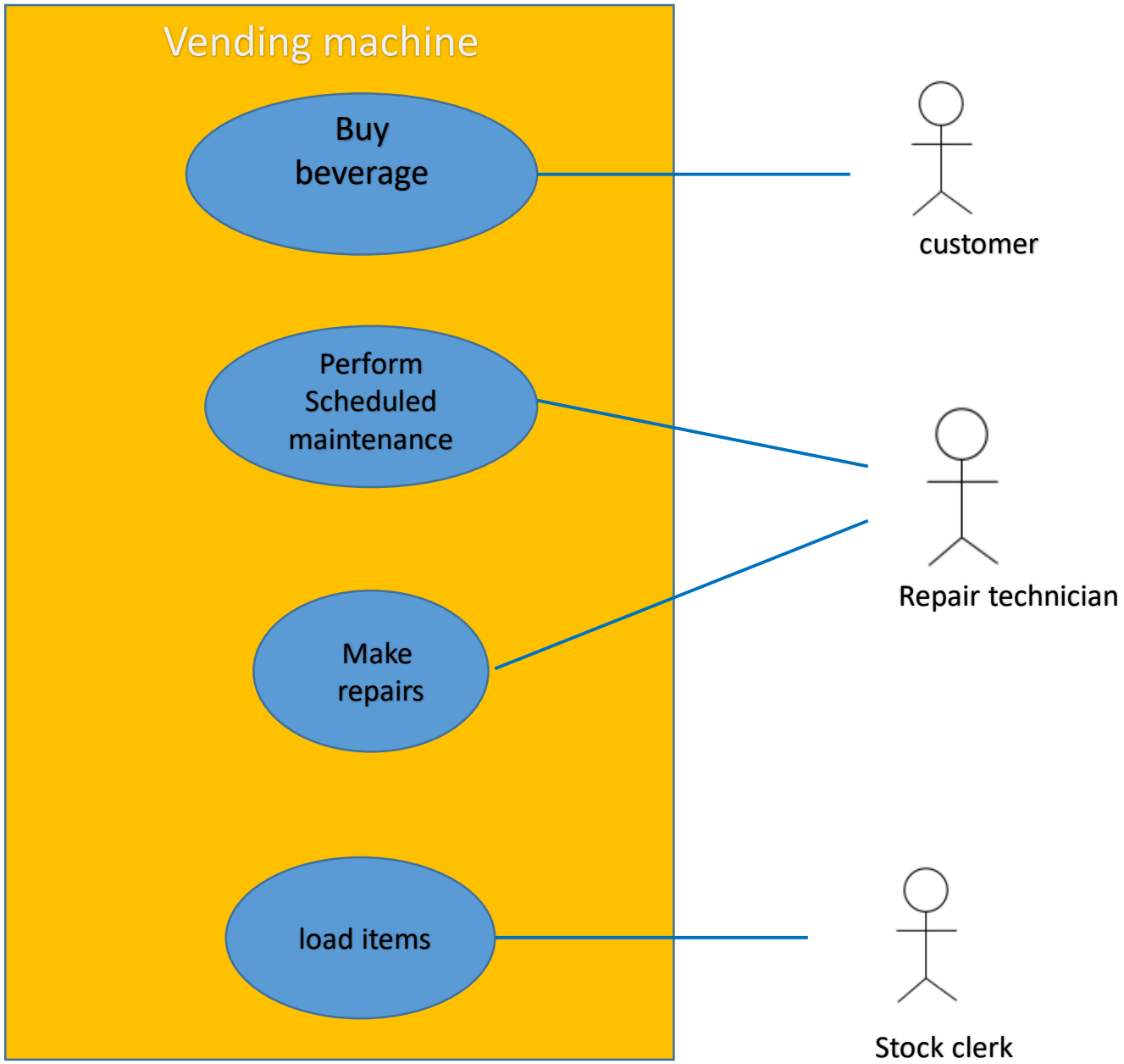
Postconditions: the machine is waiting for money to be inserted.

Use case diagram

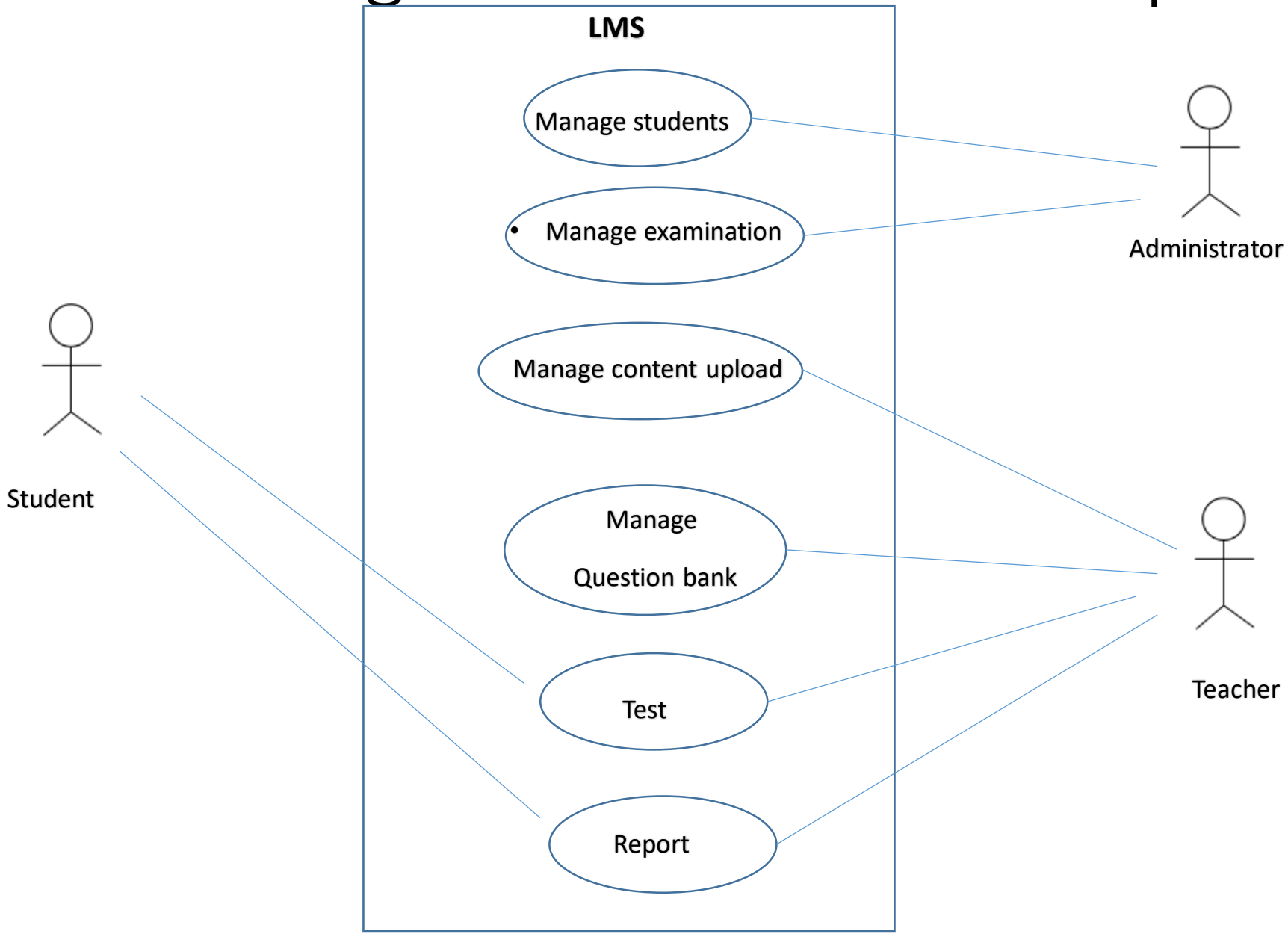
- A system involves a set of use cases and a set of actors
- Each use case represents a slice of the functionality the system provides
- Set of use cases shows the complete functionality of the system at some level of detail
- Each actor represents one kind of object for which the system can perform behavior
- The set of actors represents the complete set of objects that the system can serve

UML notations

- A **rectangle** contains the use cases for a system with actors listed on the outside
- **Name of the system** may be written near a side of the rectangle
- Use caes represented by using ellipse- name should be written within it
- A **stick man** icon denotes an actor
- **Solid lines** connect use cases with an actor


Vending machine
Buy beverage
Perform Scheduled maintenance
Make repairs
load items
customer
Repair technician
Stock clerk

Use case diagram for student LMS portal



Guidelines for use case model

- ❖Use cases identify the functionality of a system and organize it according to the perspective of users
- ❖Use cases describe complete transactions
- ❖There are some guidelines constructing the use model

- First determine the system boundary
- Ensure that actors are focused
- Each use case must provide value to users
- Relate use cases and actors
- Remember that use cases are informal
- Use cases can be structured

- First determine the system boundary

  it is impossible to identify use cases or actors if the system boundary is unclear

- Ensure that actors are focused

  -Each actor should have a single coherent purpose

  -if any object has multiple purpose , then capture them with separate actors

Example: owner of personal computer – install software, Set up a database, and

send mail

System administrator, database administrator and computer user

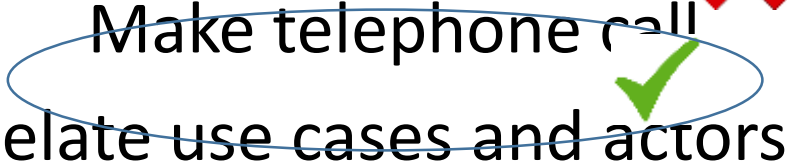
- Each use case must provide value to users

  a use case should represent a complete transaction that provides value to users and should not be defined too narrowly

Example:

Dial a telephone number

Make telephone call



- Relate use cases and actors

  Every use case should have at least one actor and every actor should participate in at least one use case

- Remember that use cases are informal
  - use case need not be formal
  - use cases are identified based on system functionality from the user centered point of view
- Use cases can be structured
  - for many applications use cases are completely distinct

Sequence Models

- The sequence model elaborates the themes of use cases
- There are two kinds of sequence models
  - scenarios
  - sequence diagram

scenarios

- A scenario is a sequence of events that occurs during one particular execution of a system
- The scope of scenario can vary
- A scenario can be the historical record or experiment of executing a proposed system
- A scenario can be displayed as a list of text statements

- Example : online food order system

John Lewis logs in
System establishes secure communications
System displays Menu
John Lewis enters a buy order for 25 biriyani
System verifies the sufficient item
System displays confirmation screen
John confirms the order and make payment
System places order
System sends information about food delivery
John Lewis logs out
System establishes insecure communication
Food delivered to the specified address

Example: juice vending machine

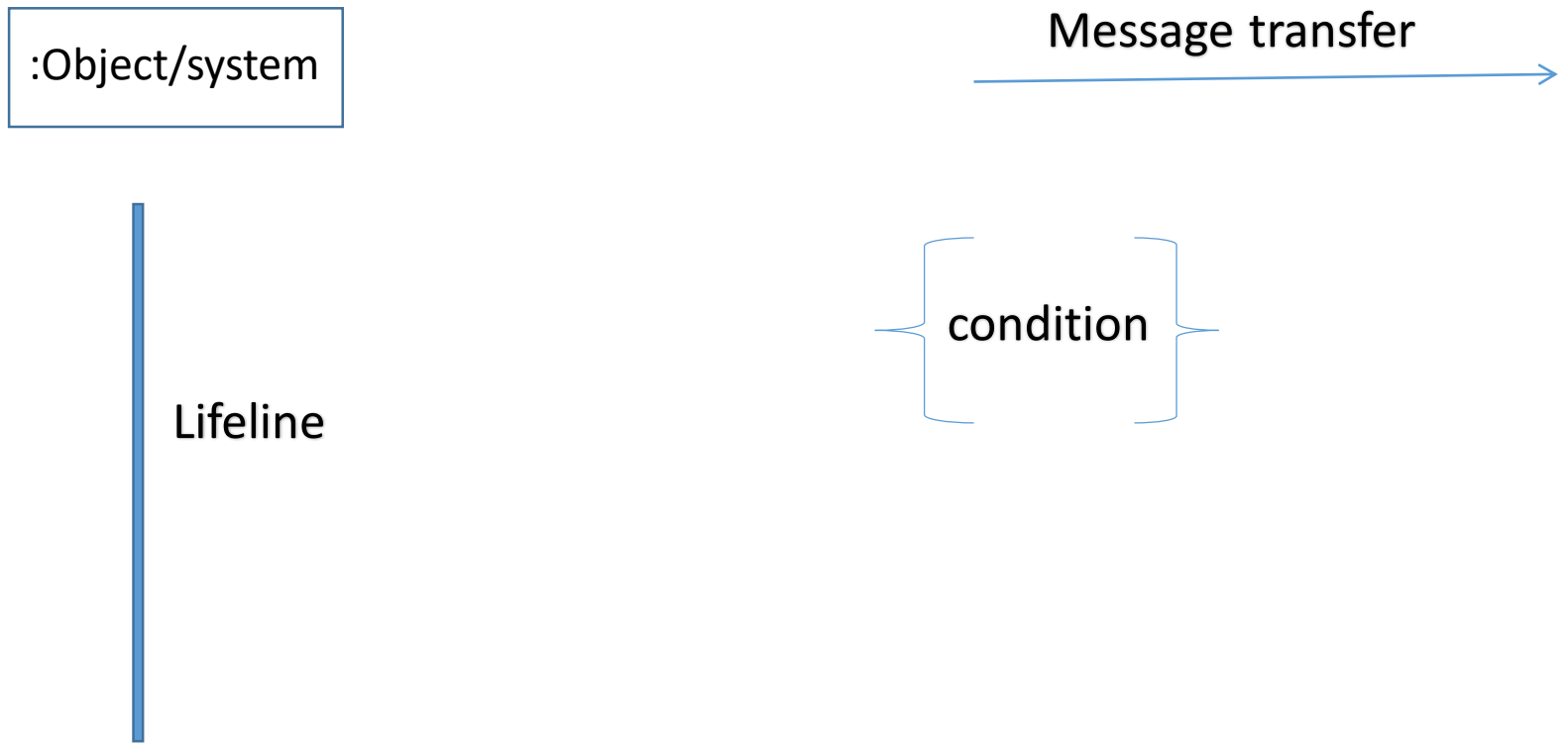
Machine displays enter coins
David Jack inserts coin into the machine
Machine displays total amount and Juice menu
David Jack selects the Juice
Machine dispenses the order
David Jack collect the juice
Machine returns change if any

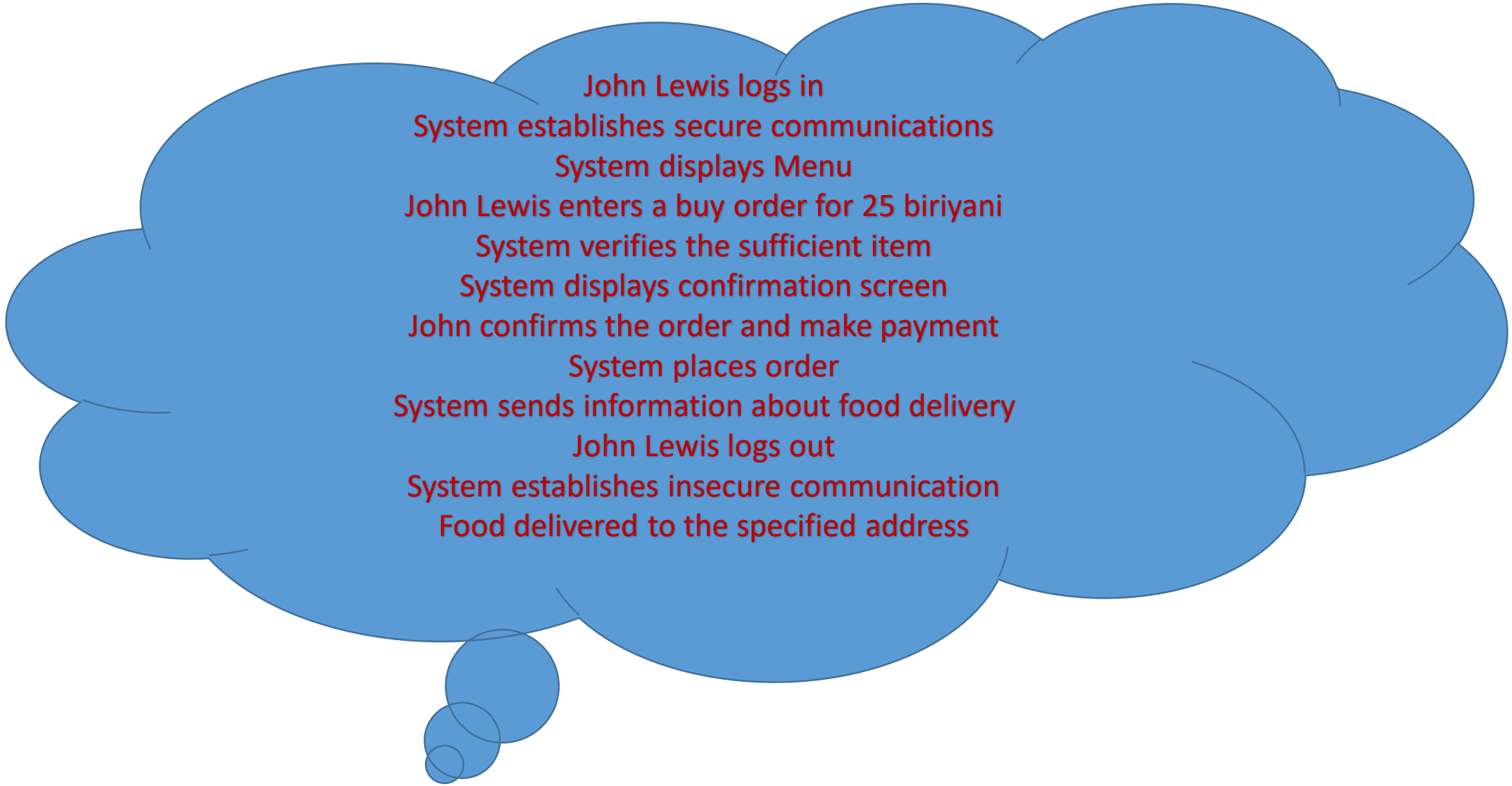
Sequence Diagram

- A sequence Diagram shows the participants in an interaction and the sequence of messages among them
- Sequence diagram shows the interaction of a system with its actors to perform all or part of a use case
- Each use case requires one or more sequence diagrams to describe its behavior
- Each sequence diagram shows a particular behavior sequence of the use case
- It is best to show specific portion of a use case

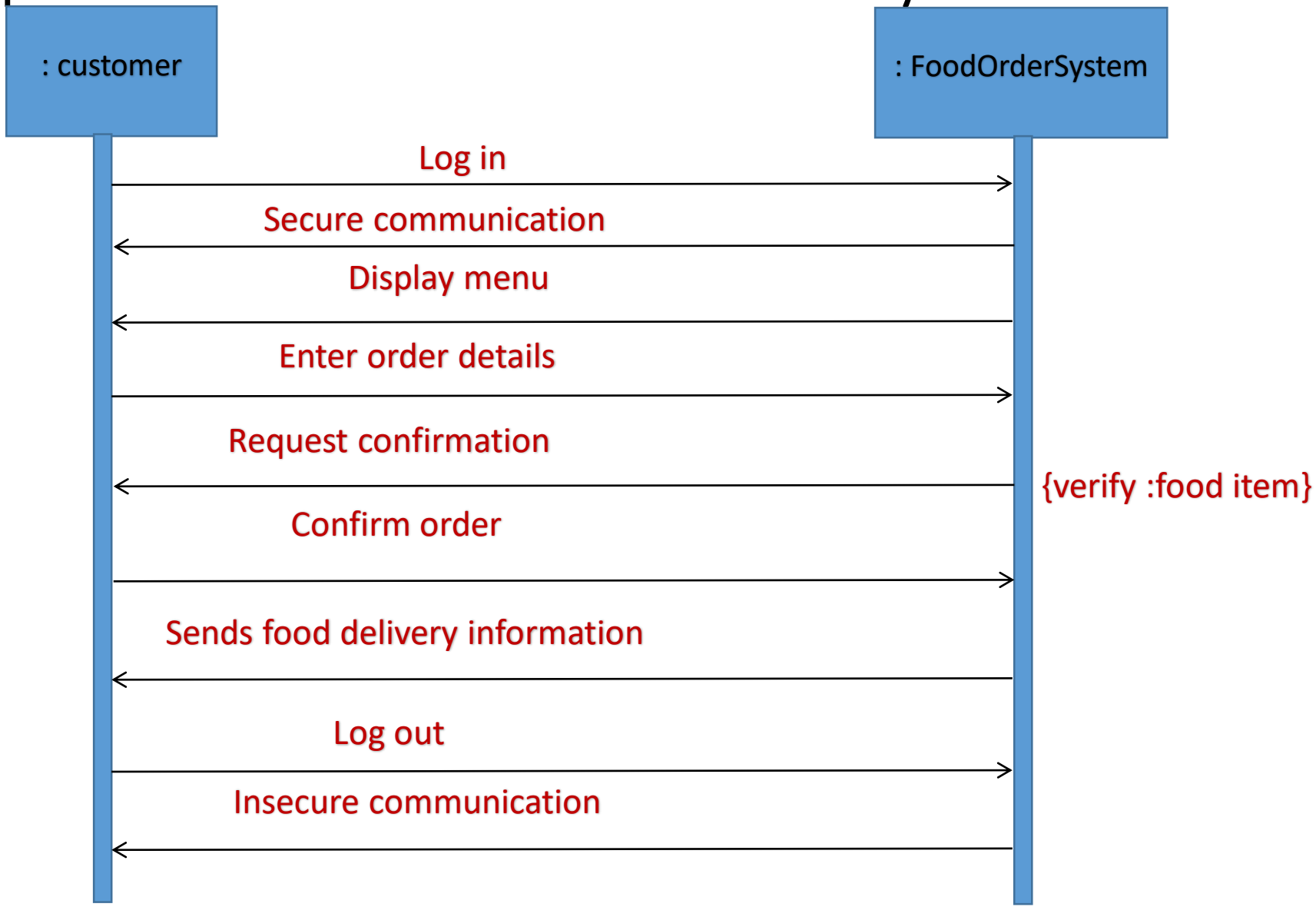
UML Notations for sequence diagram

- Actor and system is represented by a **vertical line** called a **lifeline**
- Message transfer is represented by **horizontal arrow**

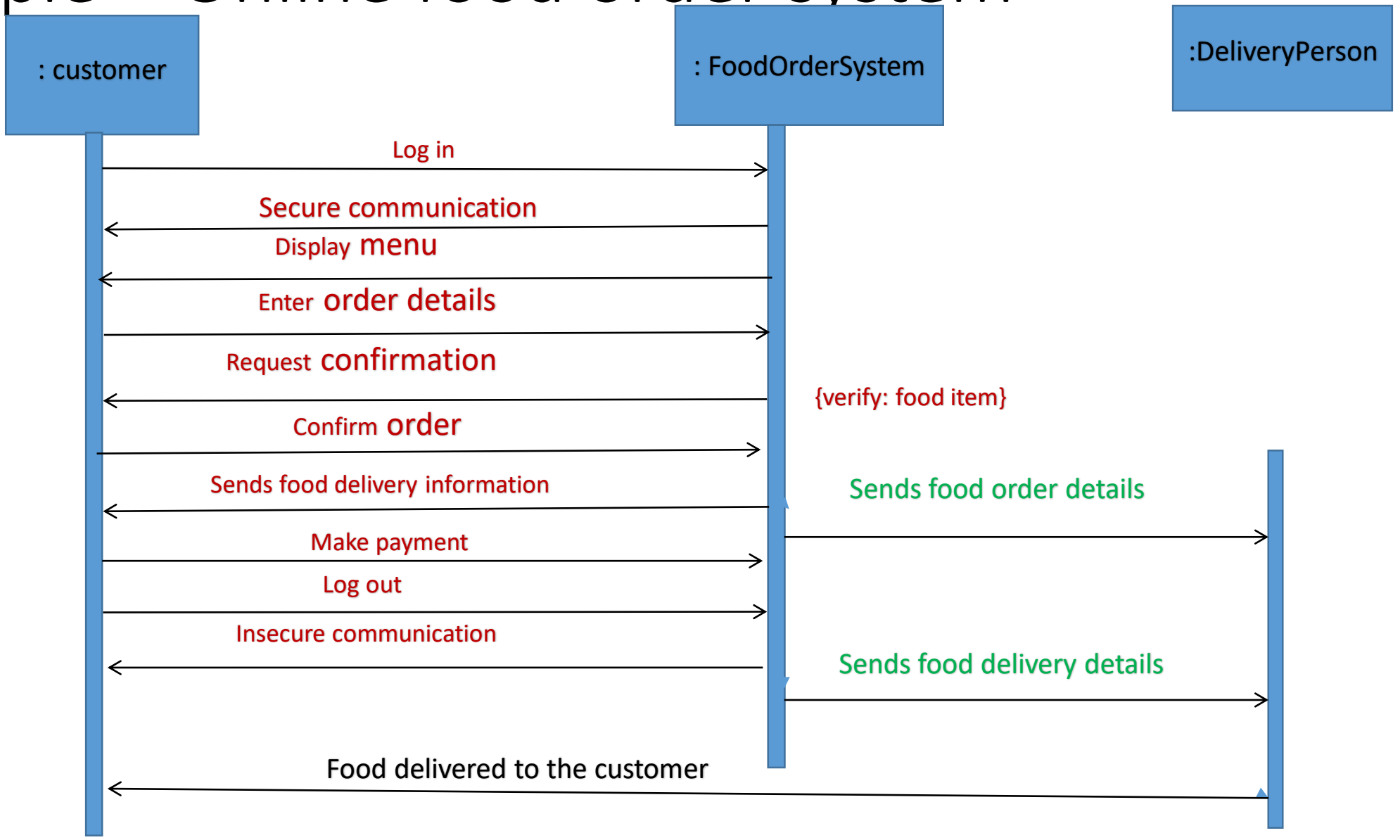

:Object/system
Message transfer
Lifeline
condition


John Lewis logs in
System establishes secure communications
System displays Menu
John Lewis enters a buy order for 25 biriyani
System verifies the sufficient item
System displays confirmation screen
John confirms the order and make payment
System places order
System sends information about food delivery
John Lewis logs out
System establishes insecure communication
Food delivered to the specified address

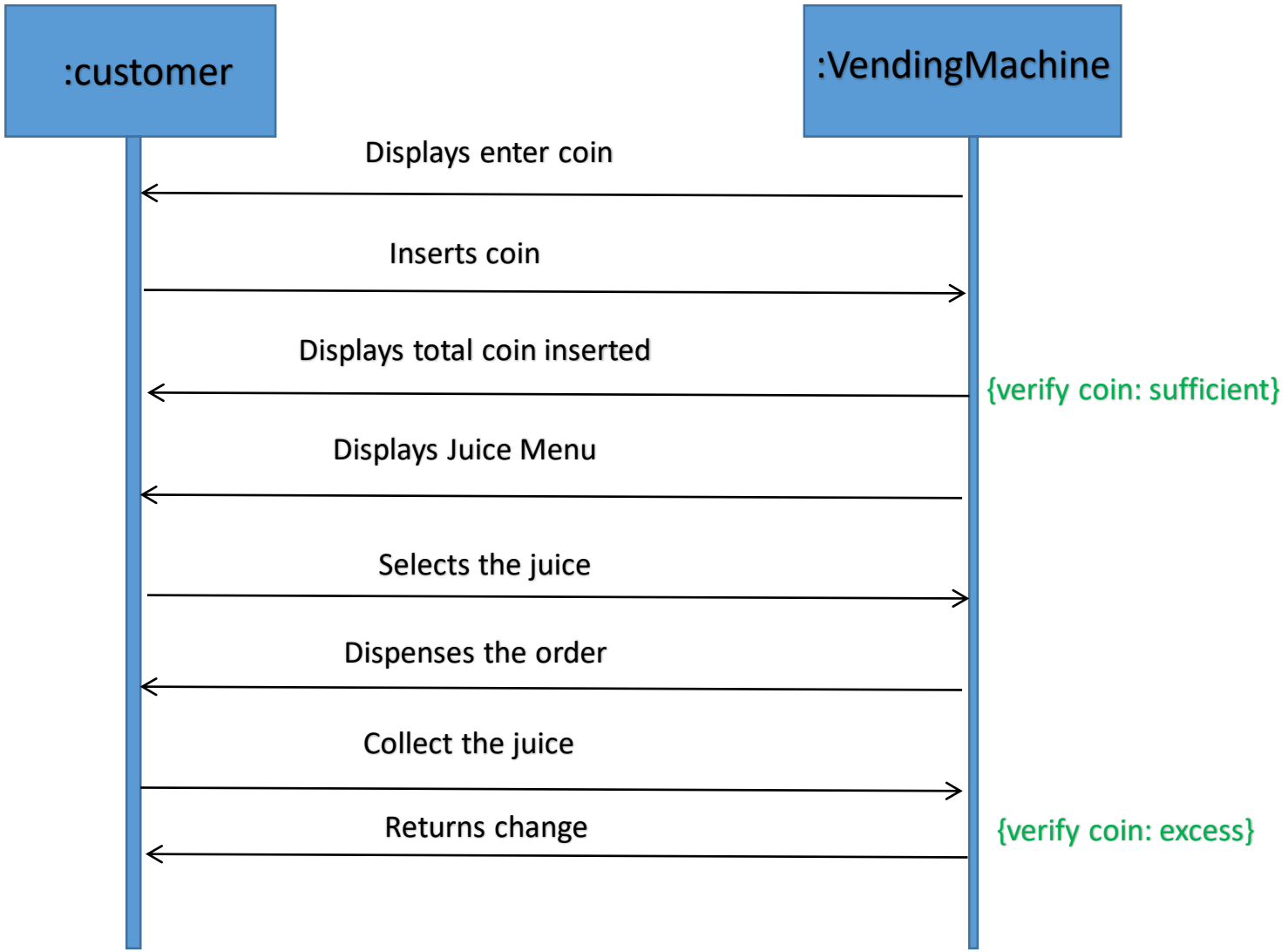
Example – Online food order system



Example – Online food order system



Example : Juice vending machine



Use Case: Buy a beverage

Summary: The vending machine delivers a beverage after a customer selects and pays for it

Actors: Customer

Preconditions: The machine is waiting for money to be inserted

Description: The machine starts in the waiting state in which it displays the message " Enter coins" A customer inserts coins into the machine. The machine displays the total value of money entered and lights up the buttons for the items that can be purchased for the money inserted. The customer pushes a button. The machine dispenses the corresponding item and make changes, if the cost of the item is less than the money inserted.

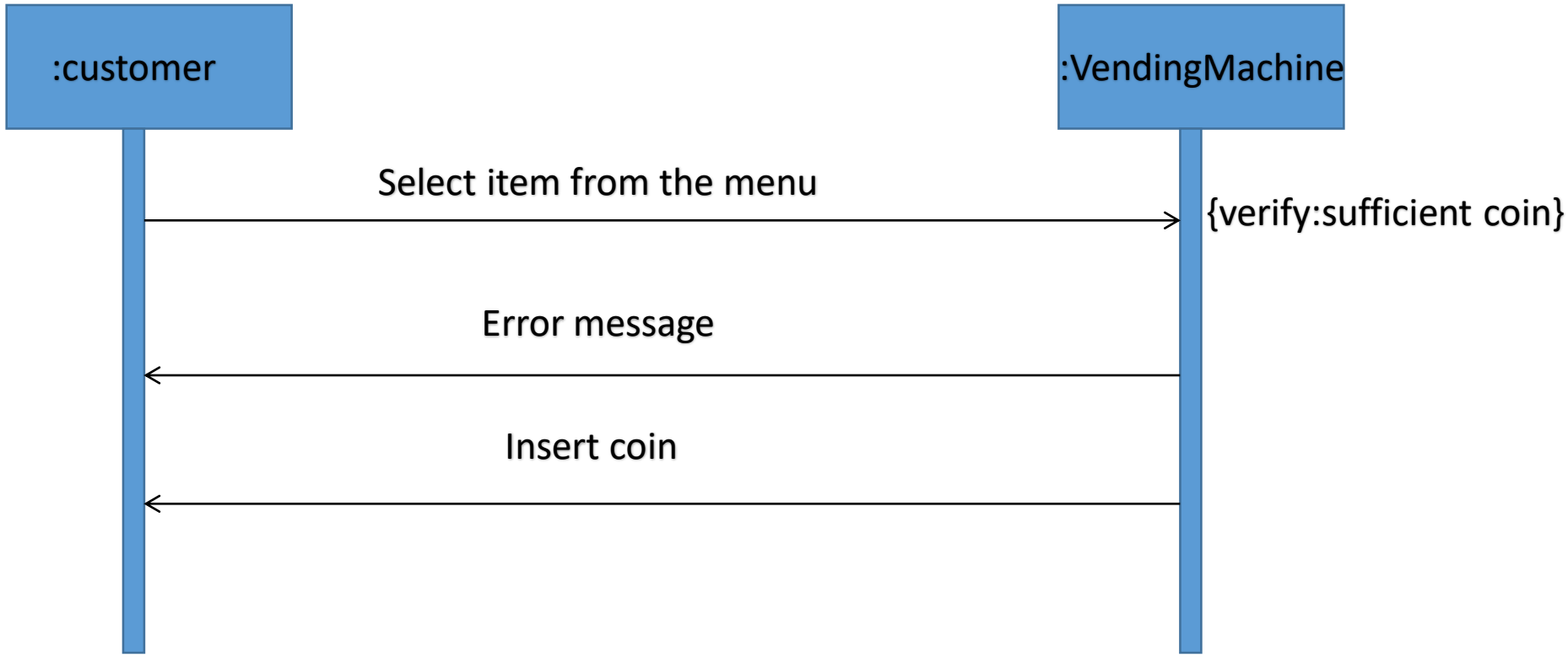
Exceptions:

Canceled: If the customer presses the cancel button before an item has been selected, the customer's money is returned and the machine resets to the waiting state.

Out of stock: if the customer presses a button for an item that costs more than the money inserted, the message "You must insert $nn.nn more for that item" is displyed, where nn.nn is the amount of additional money needed. The machine continues to accept coins or a selection

No change: if the customer has inserted enough money to buy the item but the machine cannot make the correct change, the message "Cannot make correct change" is displayed and the machine continues to accept coins or a selection.

Postconditions: the machine is waiting for money to be inserted.


:customer
:VendingMachine
Select item from the menu
{verify:sufficient coin}
Error message
Insert coin

Thank you