NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

Object Oriented Analysis & Design (CL-309)

Hamza Ahmed|| Muhammad Nadeem

hamza.ahmed@nu.edu.pk || muhammad.nadeem@nu.edu.pk

Lab Session # 07

Objectives: To Understand Interaction Diagrams (Sequence Diagram, System Sequence Diagram and Collaboration Diagram).

Interaction Diagram:

From the name Interaction it is clear that the diagram is used to describe some type of interactions among the different elements in the model. So this interaction is a part of dynamic behavior of the system.

The purposes of interaction diagrams are to visualize the interactive behavior of the system. Now visualizing interaction is a difficult task. So the solution is to use different types of models to capture the different aspects of the interaction.

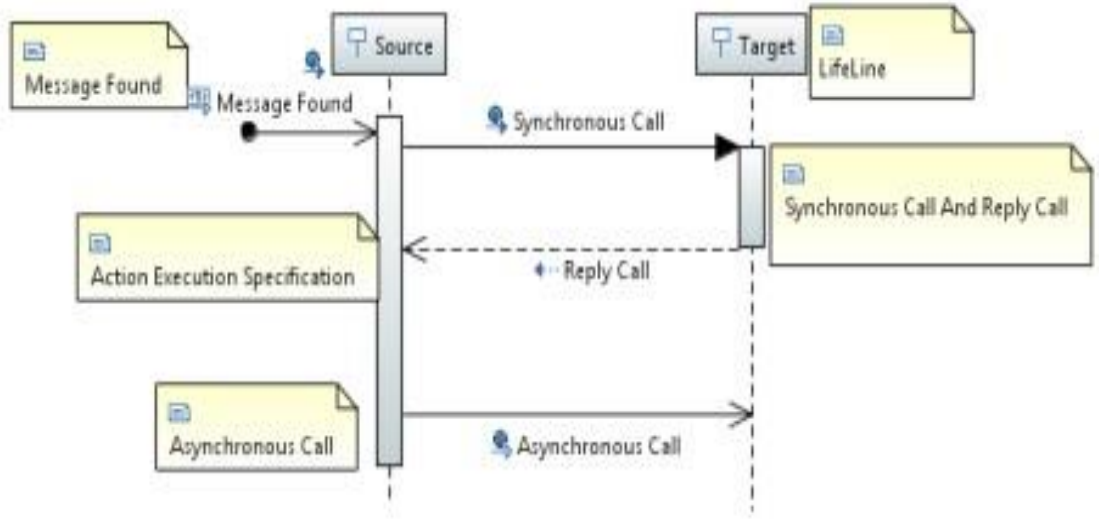


Figure 01

Combined Fragments

Demo Alt

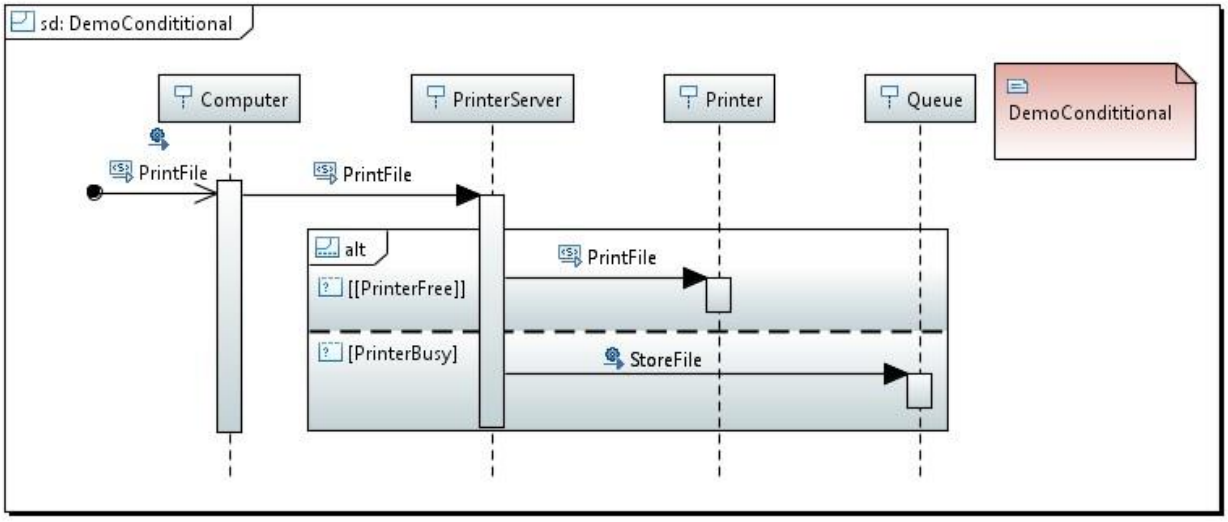


Figure 02 The messages from the PrinterServer to the Printer show how alternatives are described in a sequence diagram. Either the Print message to the Printer or the Store message to the Queue is sent.

Demo Loop

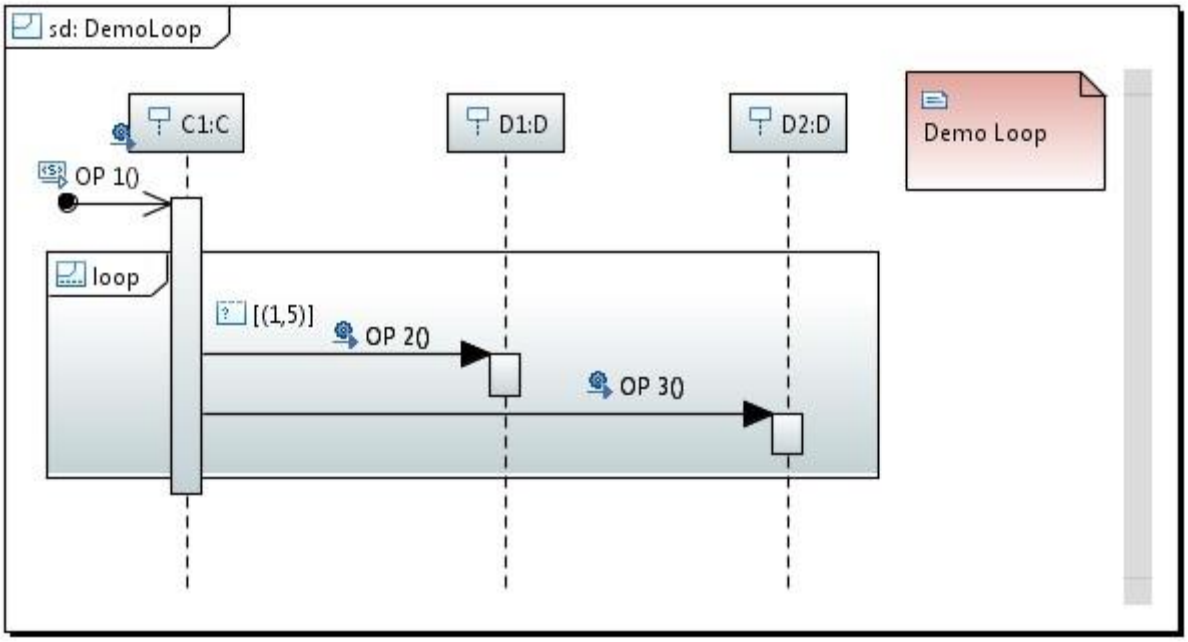


Figure 03 Iteration expressed with a loop operand.

Demo Creating Objects

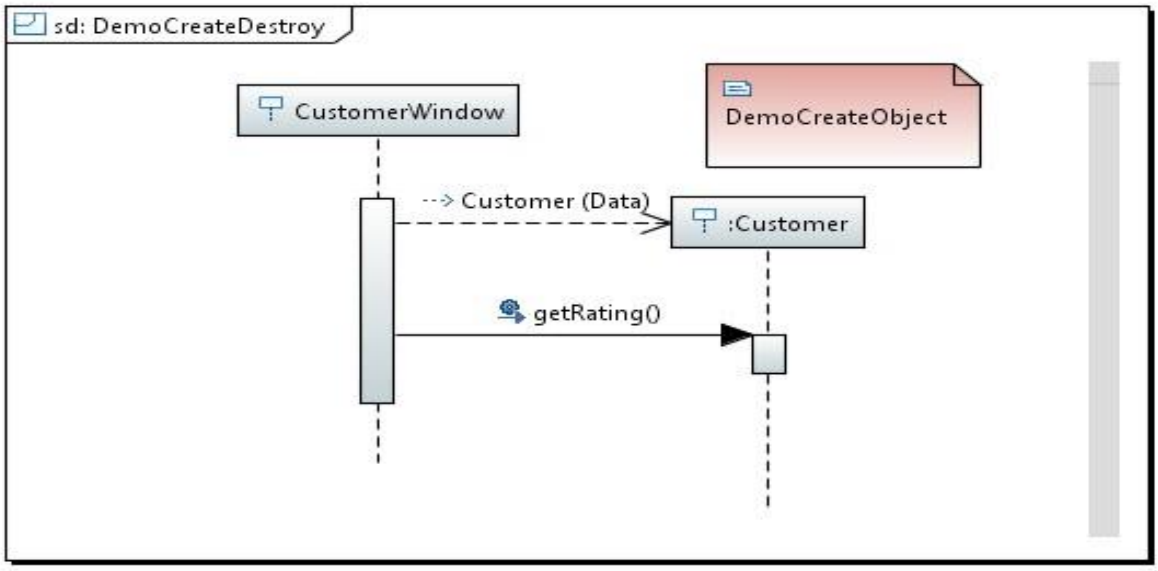


Figure 04 The Customer message creates a new object of the Customer class (this is typically handled by the constructor of the Customer class, which has the same name as the class).

Lost and Found Messages

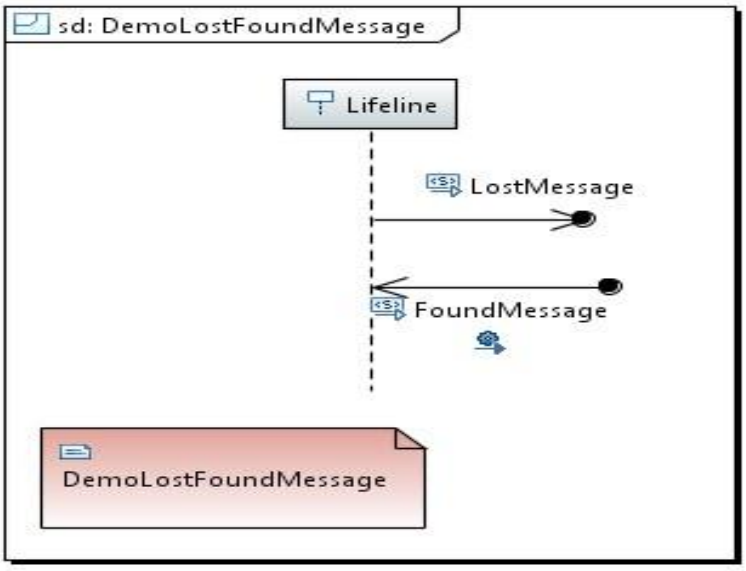


Figure 05 Lost messages are those that are either sent but do not arrive at the intended recipient, or which go to a recipient not shown on the current diagram. Found messages are those that arrive from an unknown sender or from a sender not shown on the current diagram. They are denoted going to or coming from an endpoint element.

Self Message

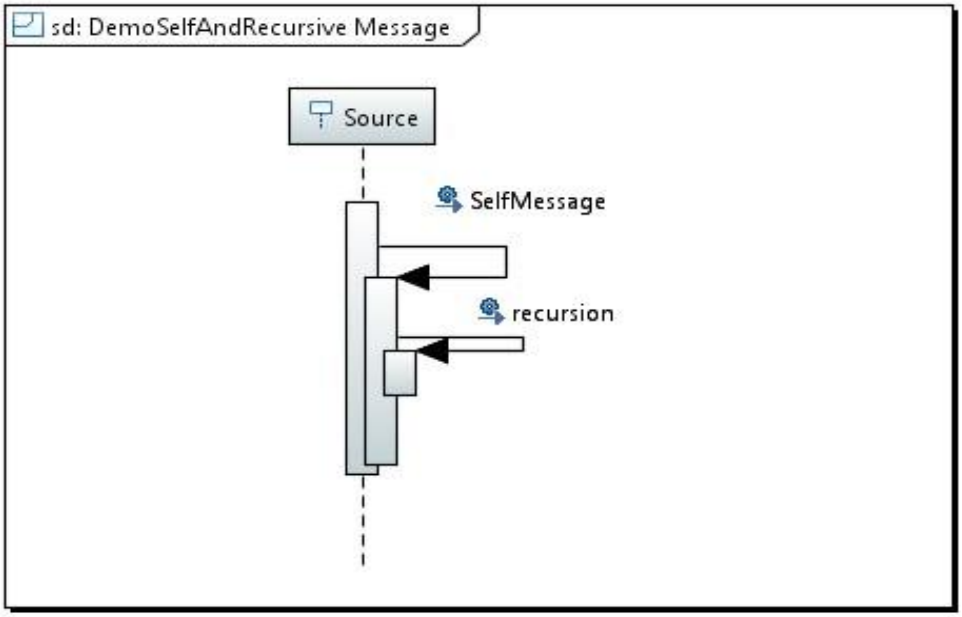
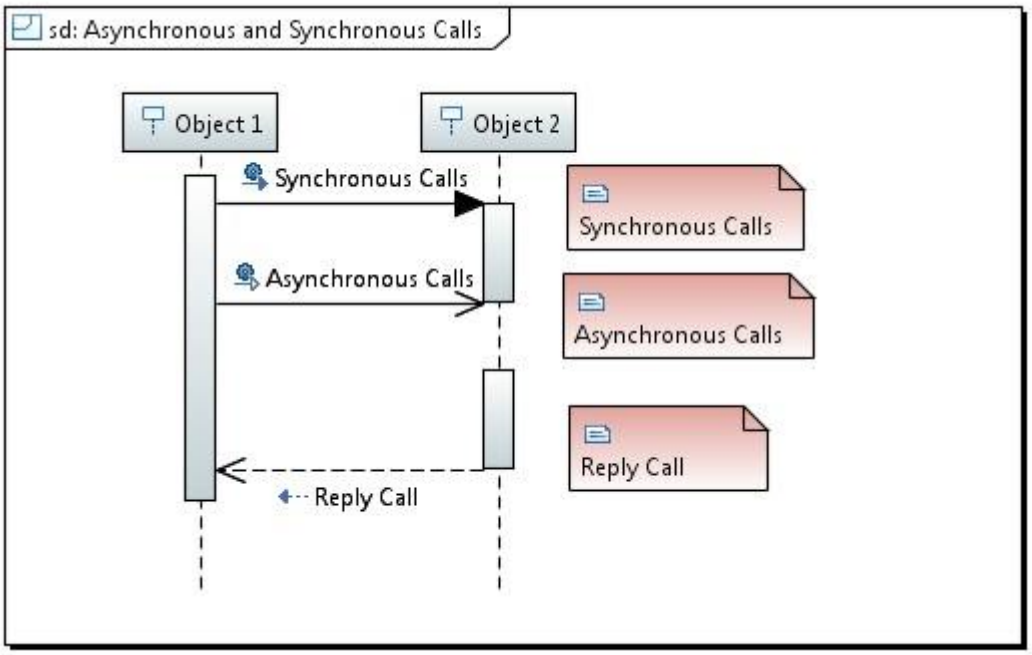
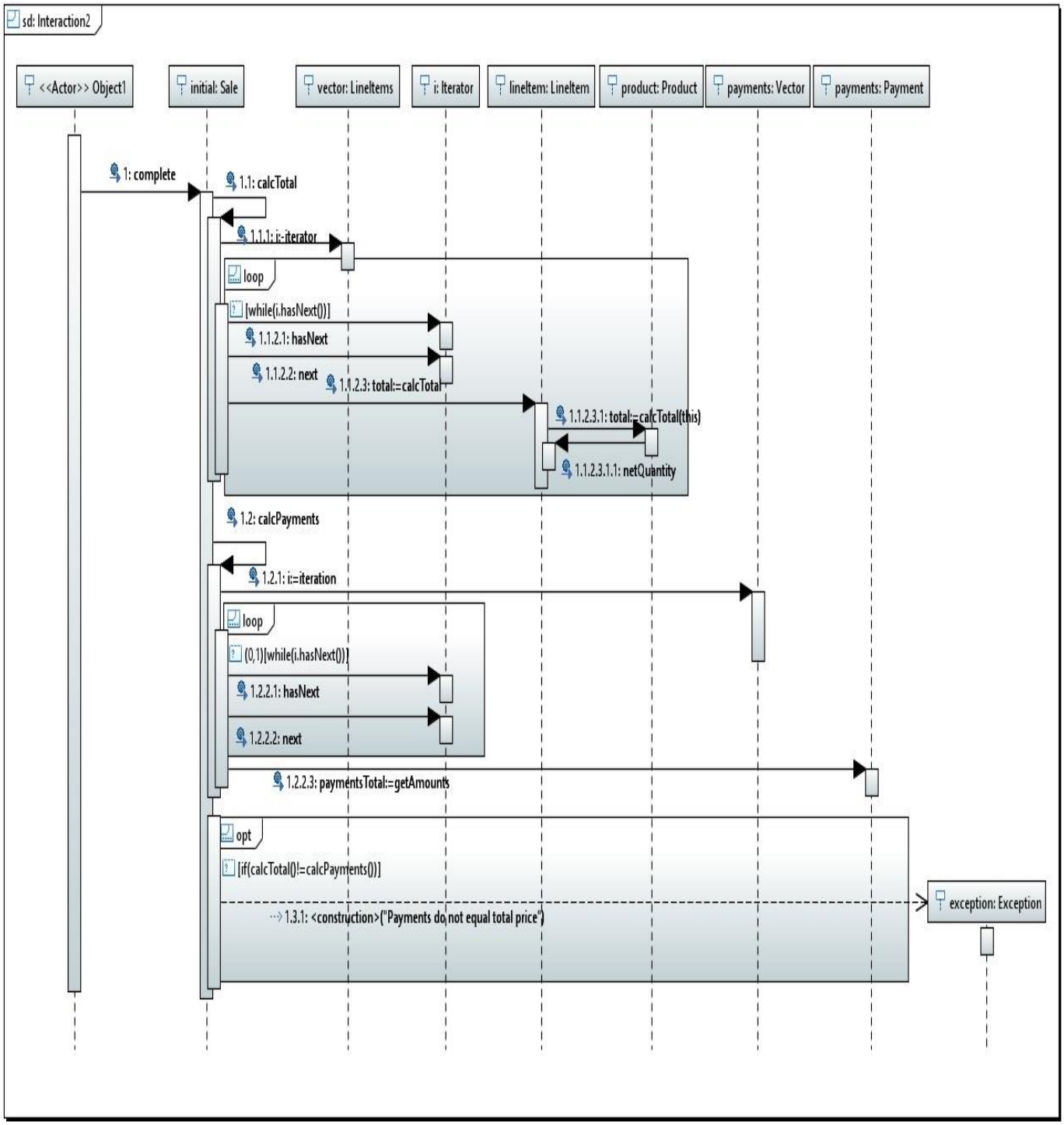


Figure 06 A self messages can represent a recursive call of an operation, or one method calling another method belonging to the same object. It is shown as creating a nested focus of control in the lifeline's execution occurrence.

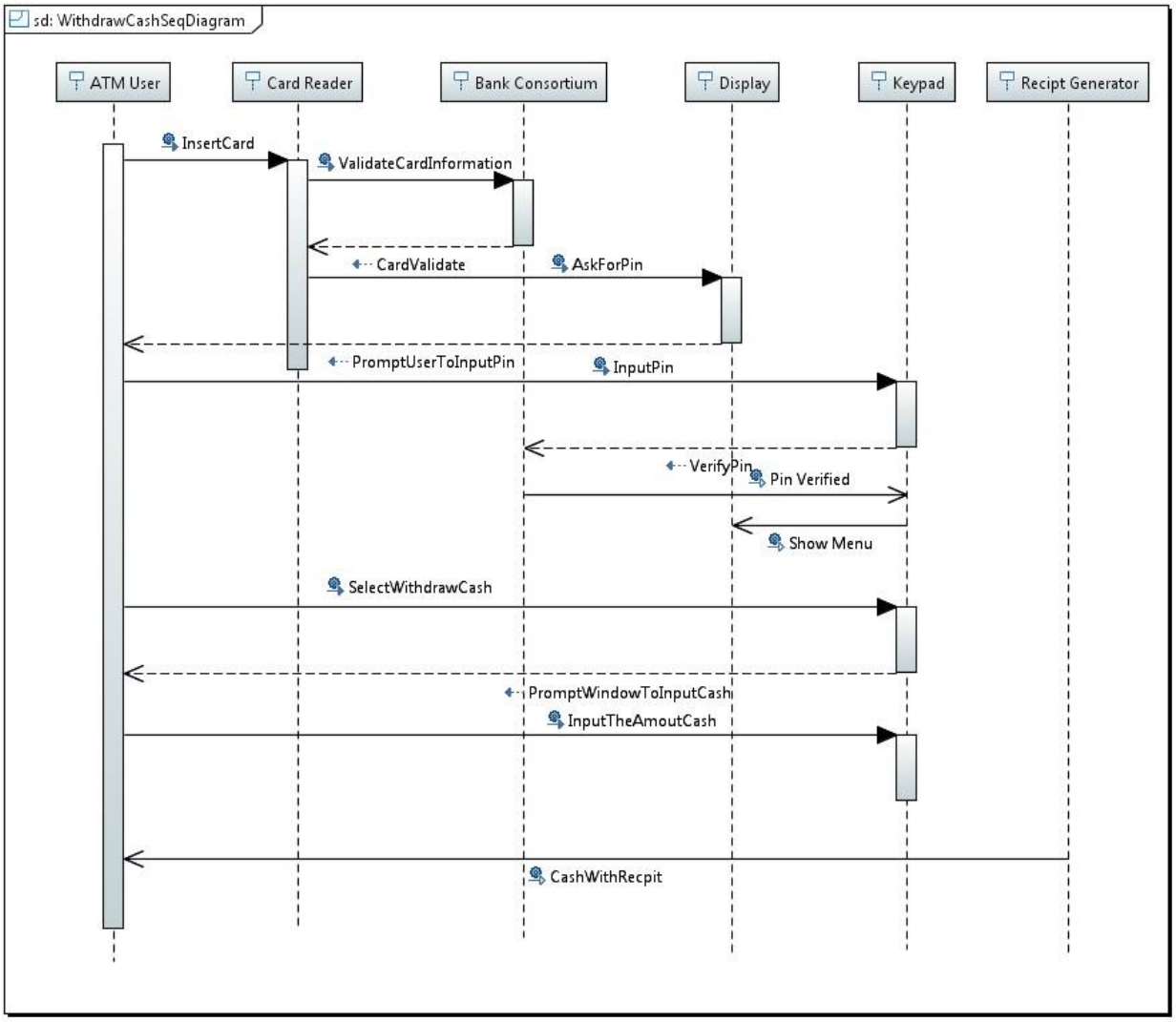
Asynchronous and Synchronous Calls



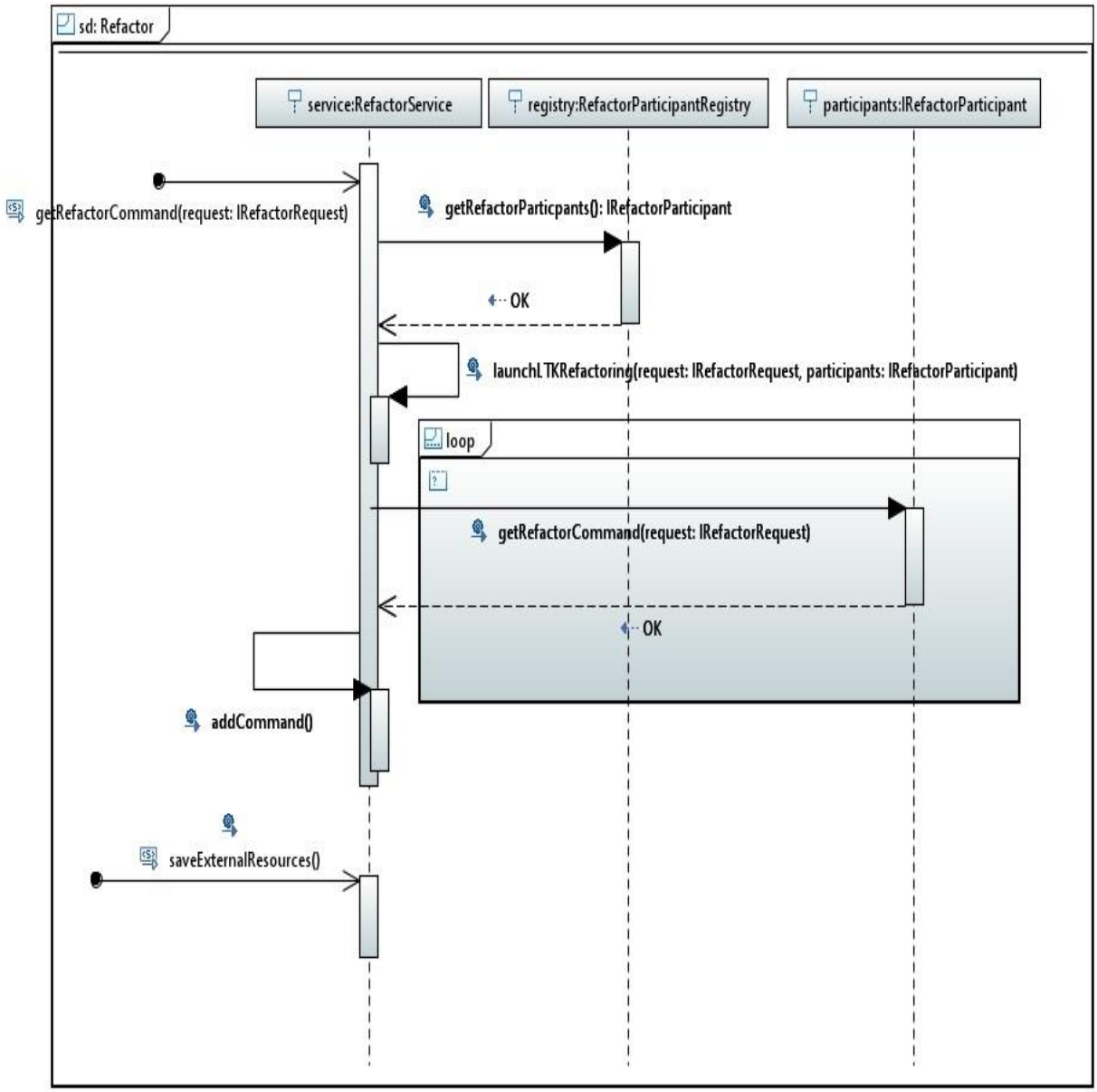
Example 01 Process Sale USE CASE



Example Withdraw Amount from ATM USE CASE



Example 03 Refactor Service



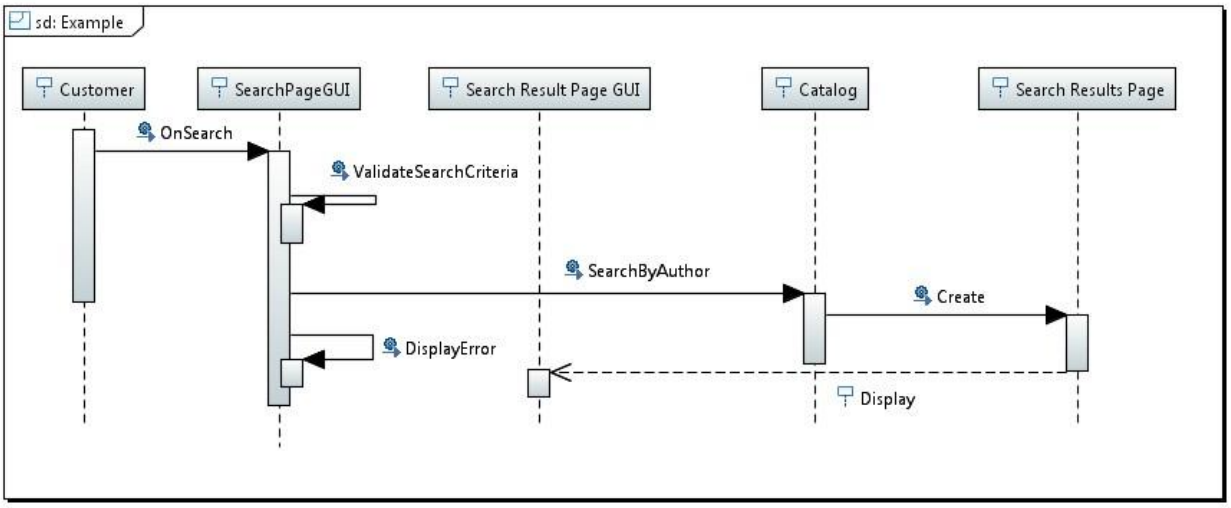
Example 4

Use Case Basic Course

The Customer specifies an author on the search page and then press the search button. The system validates the Customer's search Criteria. The system searches the catalog for books associated with the specified author. When the search is complete, the system displays the search results on the Search Results Page.

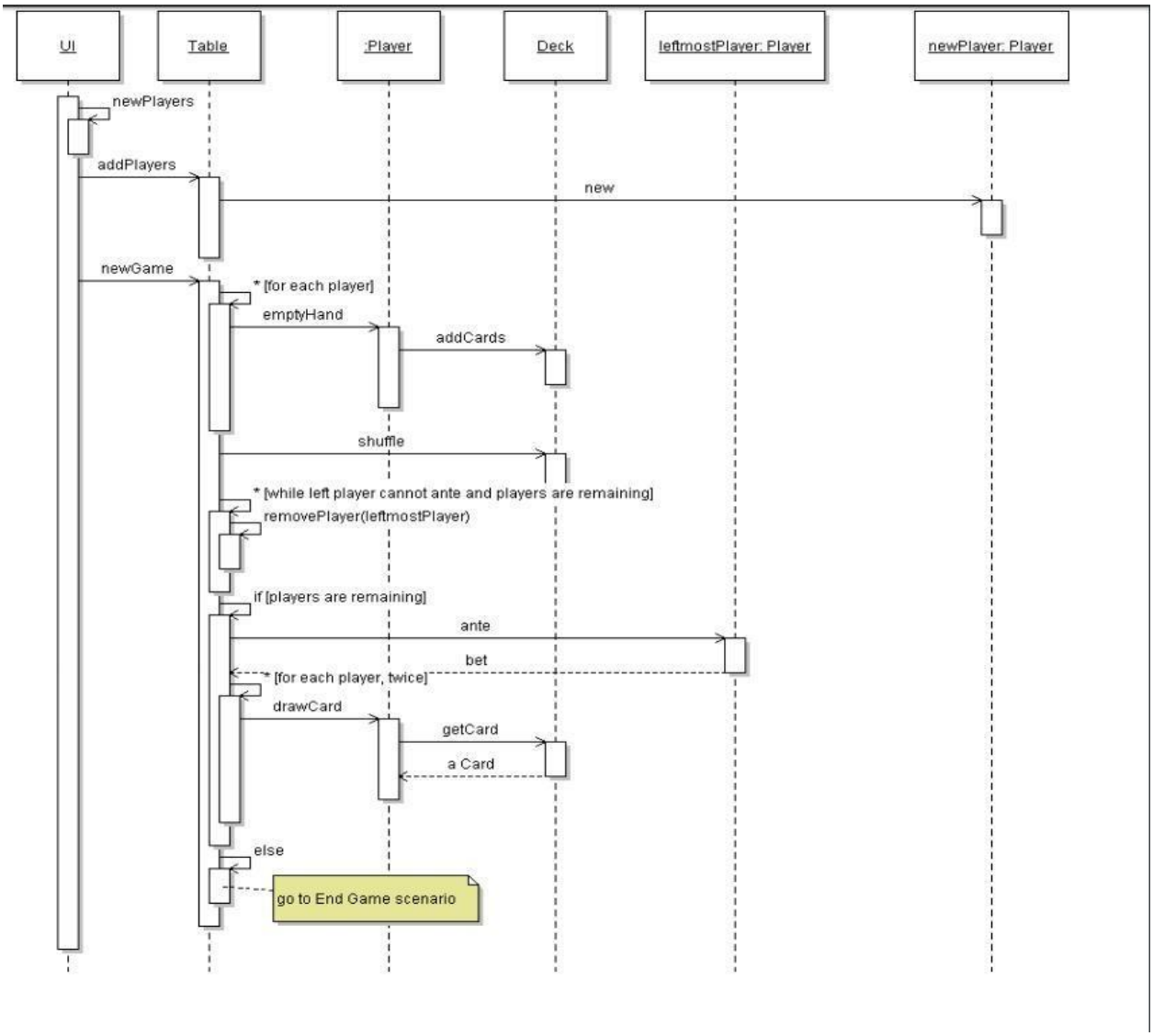
Alternate Course

If the Customer did not enter the name of an author before pressing the Search button, the system displays an error message to that effect and prompts the Customer to re-enter an author name.

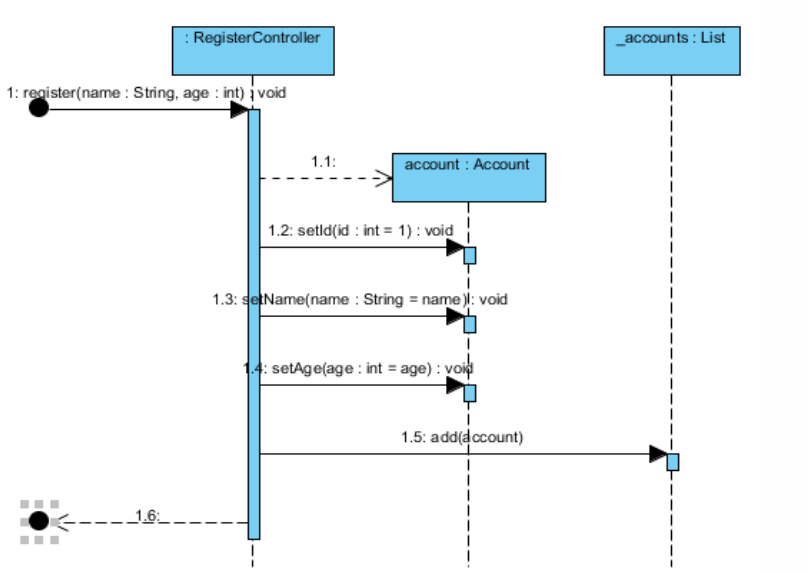


Example 5 Sequence Diagram

Let's do a sequence diagram for the following poker casual use case, Start New Game Round: The scenario begins when the player chooses to start a new round in the UI. The UI asks whether any new players want to join the round; if so, the new players are added using the UI. All players' hands are emptied into the deck, which is then shuffled. The player left of the dealer supplies an ante bet of the proper amount. Next each player is dealt a hand of two cards from the deck in a round-robin fashion; one card to each player, then the second card. If the player left of the dealer doesn't have enough money to ante, he/she is removed from the game, and the next player supplies the ante. If that player also cannot afford the ante, this cycle continues until such a player is found or all players are removed.



Example 6 Sequence diagram with Java Code




```
import java.util.*;

public class RegisterController {

  private List _accounts = new ArrayList();

        public void register(String name, int age) {
                Account account = new Account();
                account.setId(1);
                account.setName(name);
                account.setAge(age);
                _accounts.add(account);
        }

        public List getAccounts(){
                return _accounts;
        }
}
```

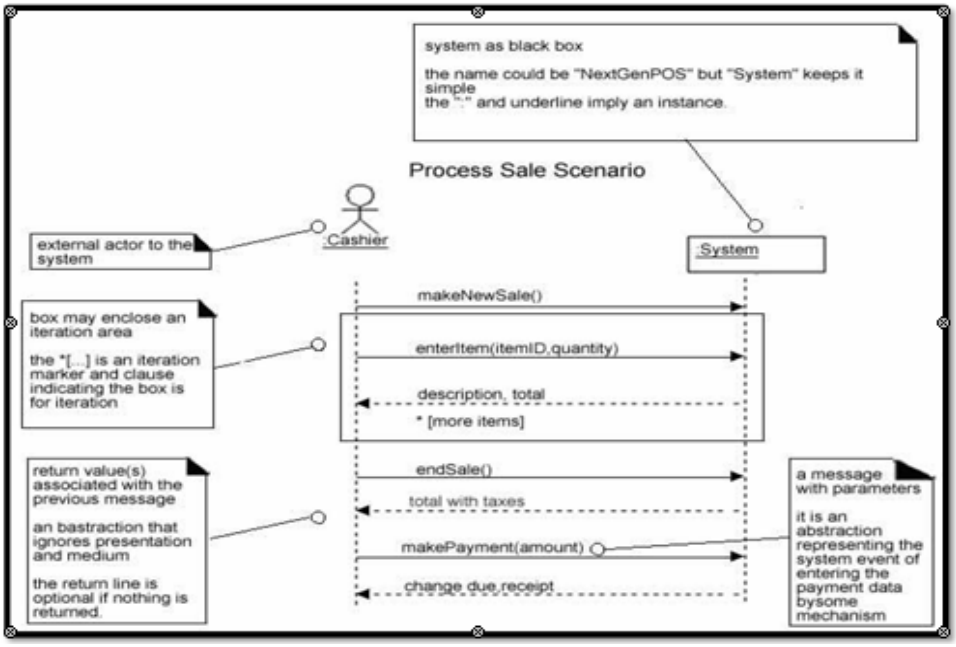
```
public class Account {

  private int _id;
  private String _name;
  private int _age;

  public void setId(int id){
    _id = id;
  }

  public int getId(){
    return _id;
  }

  public void setName(String name){
    _name = name;
  }

  public String getName(){
    return _name;
  }

  public void setAge(int age){
    _age = age;
  }

  public int getAge(){
    return _age;
  }

}
```

System Sequence Diagram Behavior

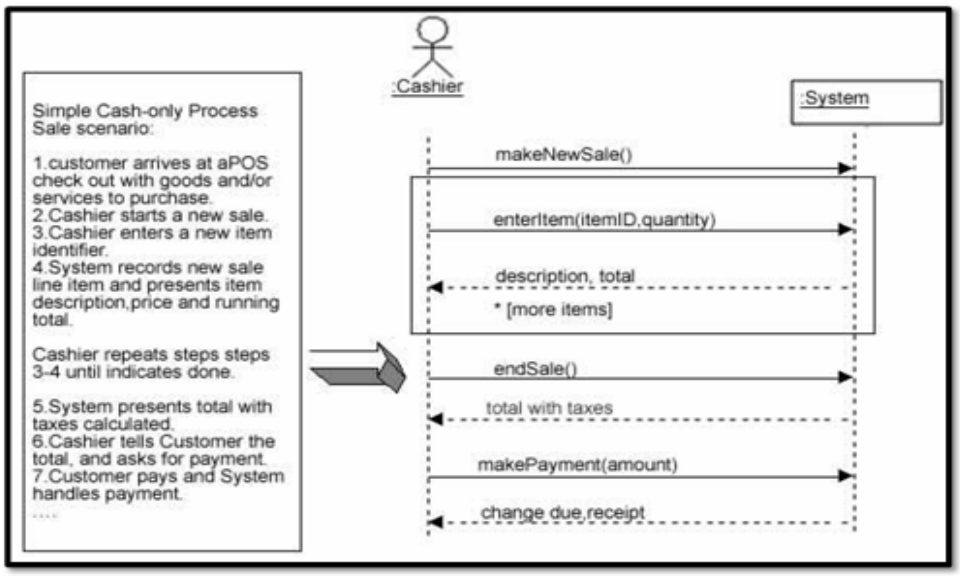
- System behaves as “Black Box”.
- Interior objects are not shown, as they would be on a Sequence.

Example of an SSD Following example shows the success scenario of the Process Sale use case. Events generated by cashier (actor)

- **makeNewSale - enterItems - endSale and - makePayment**



SSD with USE CASE text



Collaboration/ Communication Diagram

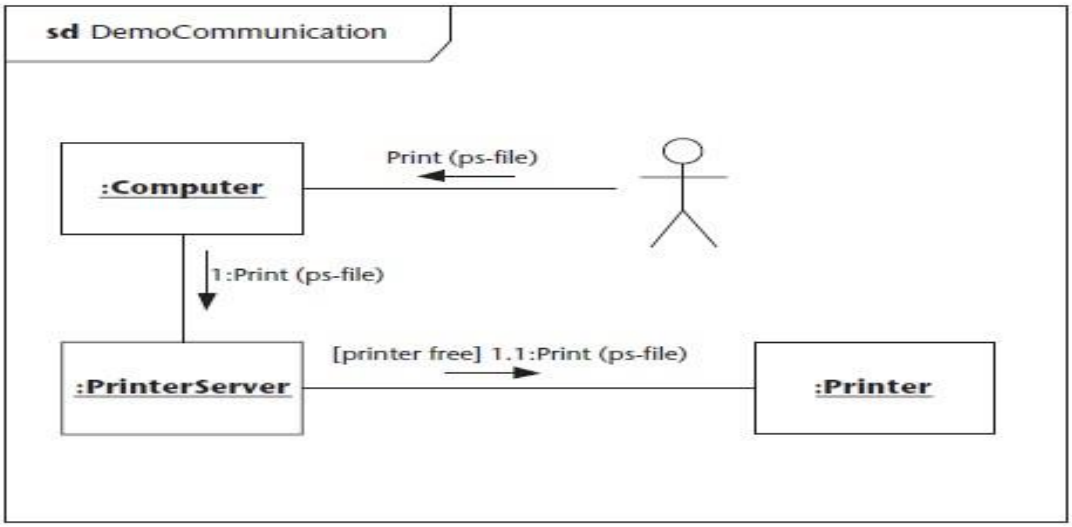


Figure-01: PrintFile communication Diagram

[An actor sends a Print message to the Computer. The Computer sends a Print message to the PrinterServer. The PrinterServer sends a Print message to the Printer if the printer is free.]

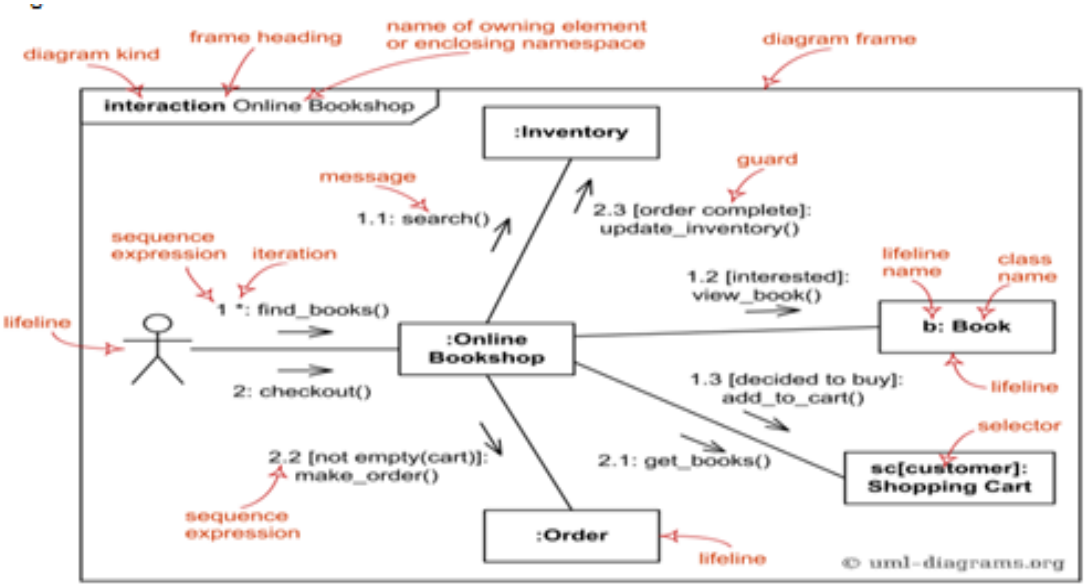


Figure-02: An example of UML communication diagram for Online Bookshop.

Communication Diagram with Java Code

```
Public class A {
        Private B myB = new B ();
Public void doOne() {
        myB.doTwo();
        myB.doThree();
}
…
}
```

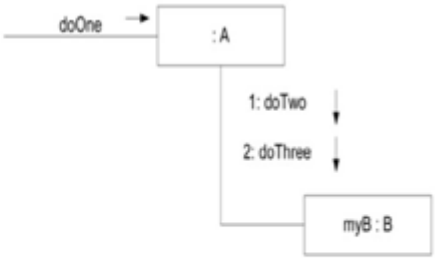


Figure 3

Figure-03: An illustration of OOP conceptual code conversion to communication diagram

Movie Storage System Use Case Diagram

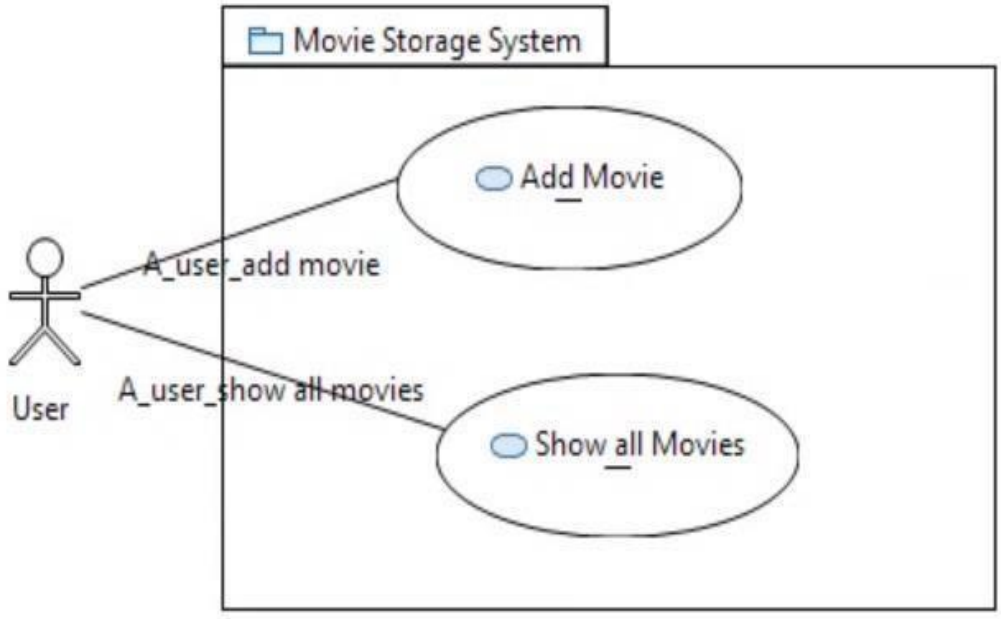


Figure-04 - An example USE CASE diagram for the "Movie Storage System"

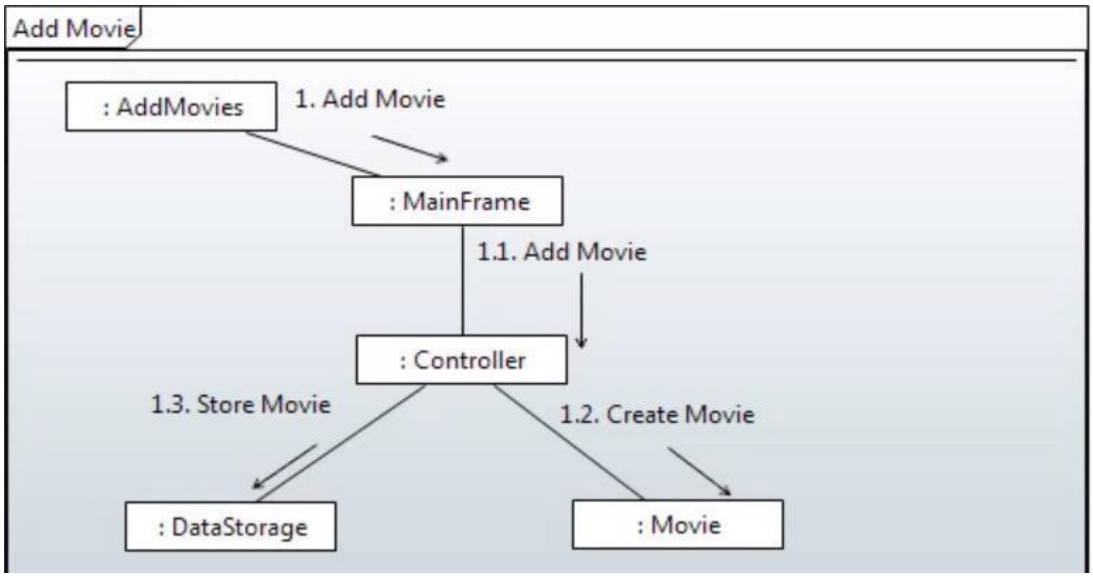


Figure-05(a) an example communication diagram for the above depicted "Movie Storage System's AddMovie use case"

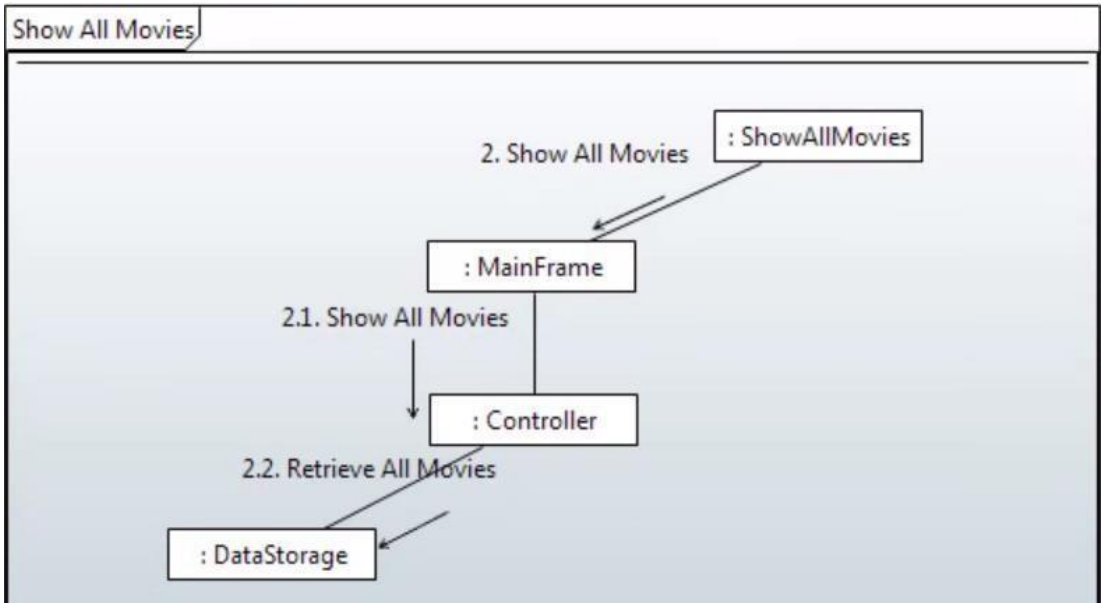


Figure-05(b) an example communication diagram for the above depicted "Movie Storage System's Show All Movies use case"