Unified Modeling Language (UML)

Object Oriented Modeling and Programming

Interaction models

Sequence Diagrams

Collaboration Diagrams

Activity – Identify the Interactions

When a Customer visits a Restaurant, they are greeted by a Waiter. The Customer places an order for food and drinks. The Waiter forwards the order to the Kitchen and Beverage Counter, respectively. The Chef prepares the food, while the Barista prepares the drinks.

Once ready, the Waiter serves the food and drinks to the Customer. After finishing, the Customer requests the bill, and the Waiter provides it. The Customer pays via Cash or Card, and the Waiter processes the payment through the Billing System.

Interaction models

- All systems involve interaction of some kind.
- There are different types of interactions;
  - User interactions-involve user inputs and outputs
  - Interactions between the software and other systems in the environment
  - Interactions between the components of a software system
- Show how a set of **actors and objects communicate with each other** to perform the steps of a use case, or of some other piece of functionality.

Interaction model

- Elements in an interaction diagram:
  - **Instances of classes or actors:** instances of classes (i.e. objects) are **shown as boxes** with the class and object identifier underlined
  - **Messages: different types of communication;** These are **shown as arrows from actor to object, or from object to object.**
- The main objective
  - **to better understand the sequence of messages.**

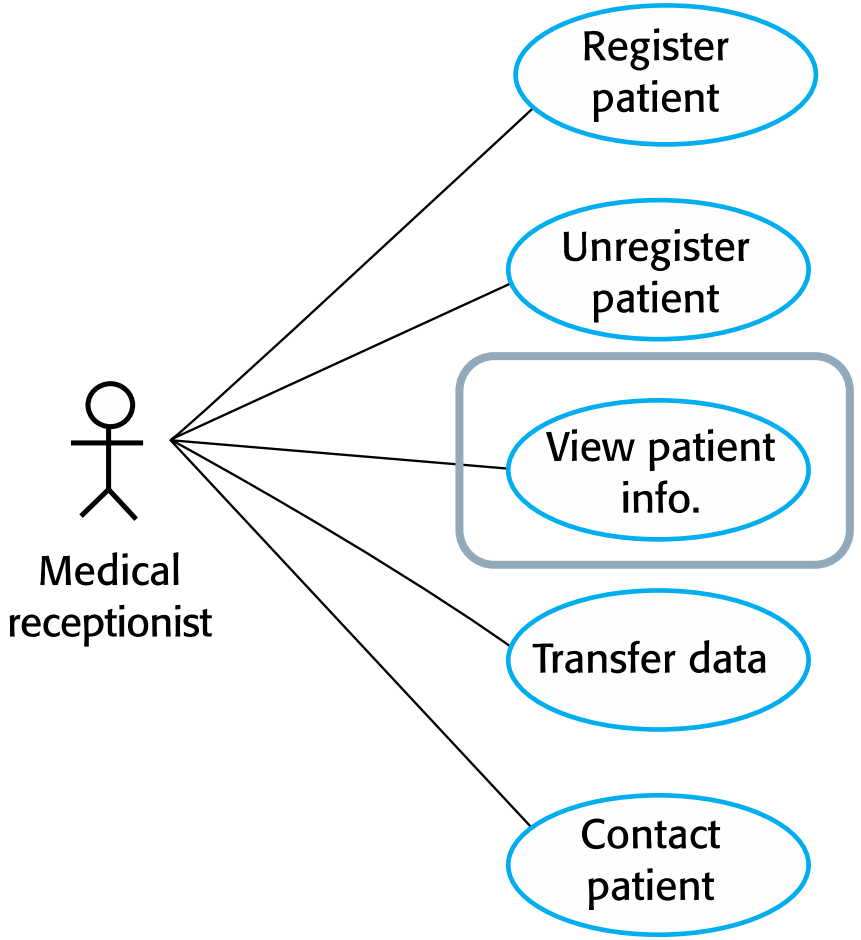
Sequence Diagrams

The interaction that takes place in a collaboration

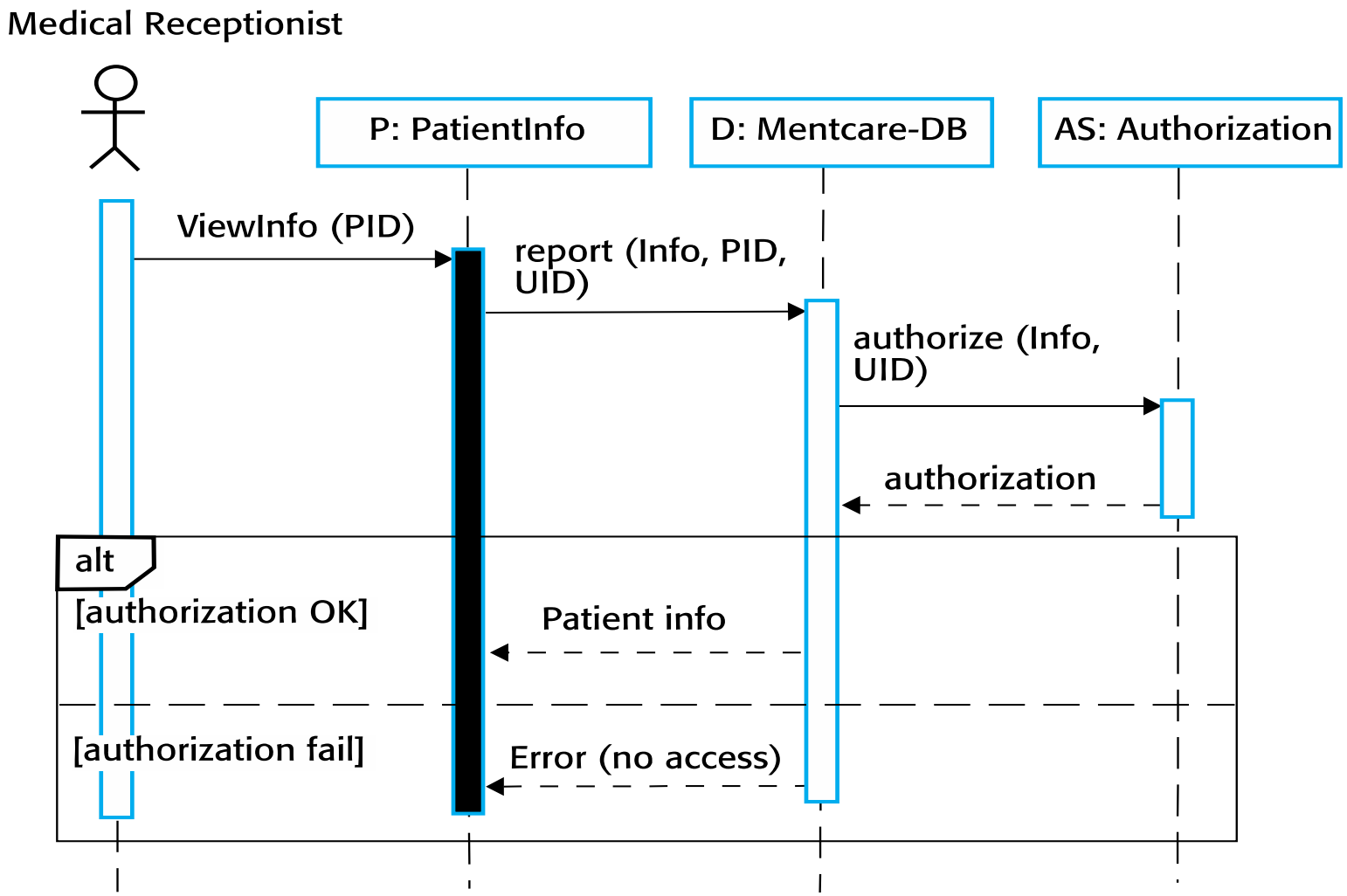
Sequence Diagrams

- Commonly used by developers.
- Model the interactions between objects in a single use case.
- Illustrate how the different parts of a system interact with each other to carry out a function, and the order in which the interactions occur when a particular use case is executed.
- Shows different parts of a system work in a ‘sequence’ to get something done.
- Shows elements as they interact over time and they are organized according to object (horizontally) and time (vertically).
- Object Dimension: The elements that are involved in the interaction
- Time Dimension: Ordering, not duration

Use case: 'Medical Receptionist' (Medical Health System)



Sequence diagram: ‘View patient infor’



Notations

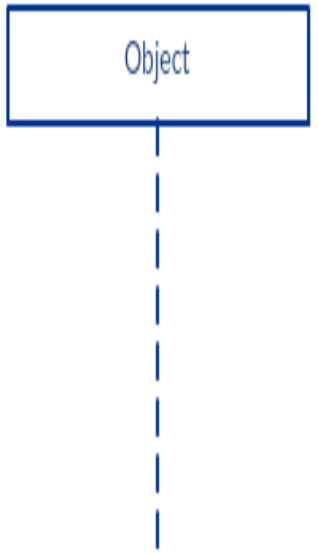
Sequence Diagram

Ref: https://creately.com/blog/diagrams/sequence-diagram-tutorial/

Basic Symbols and Components

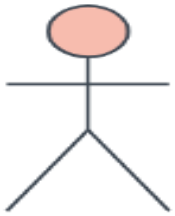
Lifeline Notation

- Represents an individual participant in the Interaction.
- Made up of several of these lifeline notations
- Should be arranged horizontally across the top of the diagram.
- No two lifeline notations should overlap each other.
- Represent different objects or parts that interact with each other in the system during the sequence.



Basic Symbols and Components

- A lifeline notation with an actor element symbol
  - Used when the particular sequence diagram is owned by a use case



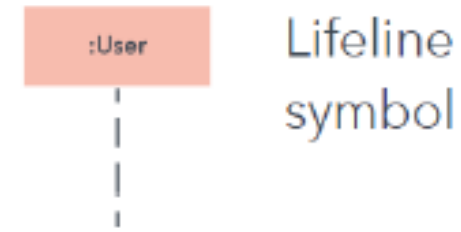
Actor symbol

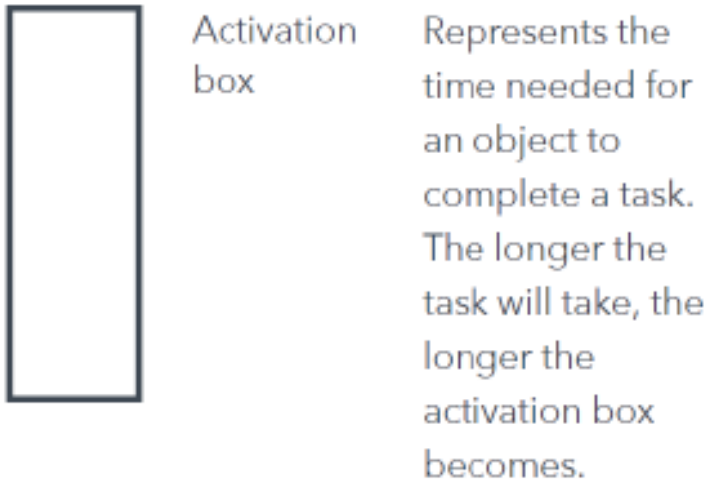
Shows entities that interact with or are external to the system.

Object symbol

Represents a class or object in UML. The object symbol demonstrates how an object will behave in the context of the system.

Basic Symbols and Components





- Lifelines begin with a labeled rectangle shape or an actor symbol.
- Represents the passage of time as it extends downward.
- This dashed vertical line shows the sequential events that occur to an object during the charted process.

Basic Symbols and Components

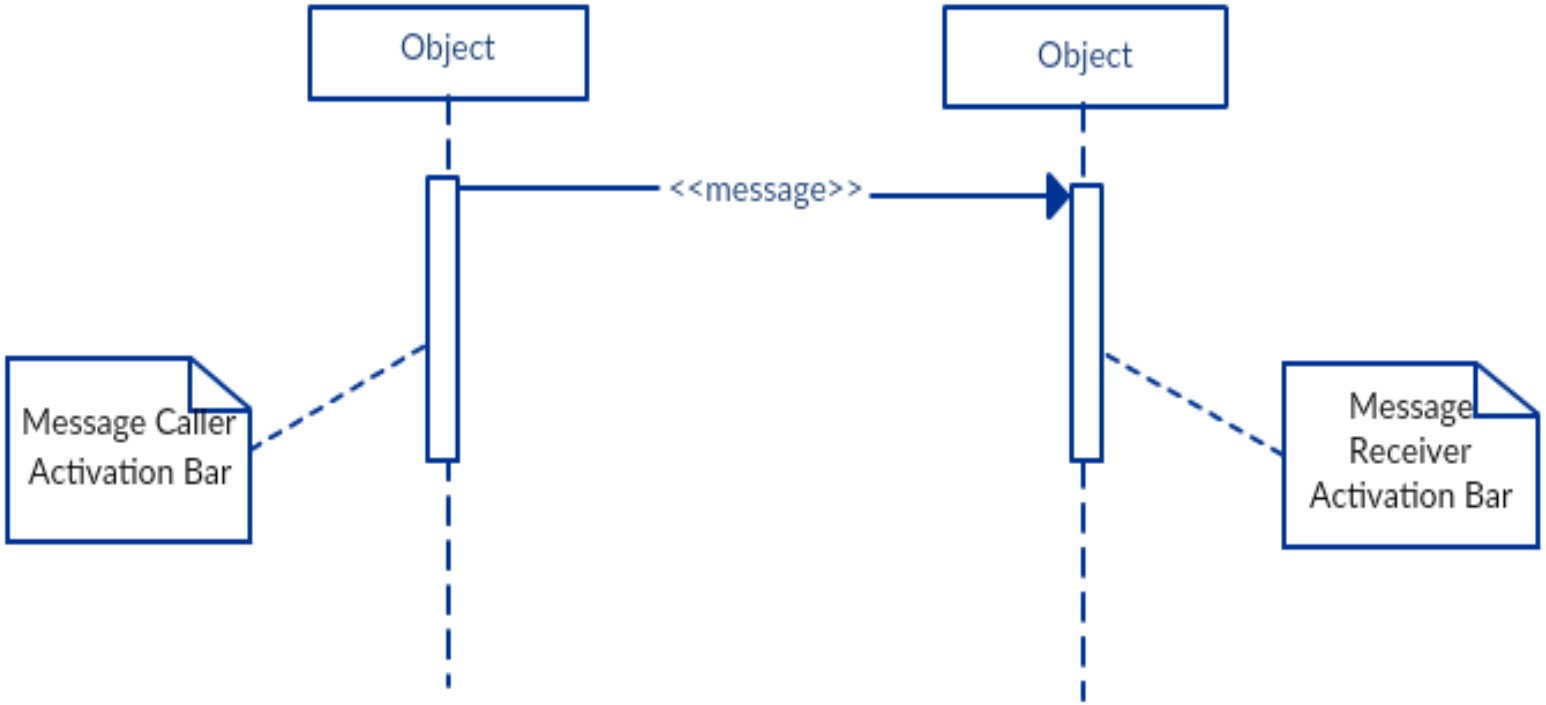
Activation Bars

- A box placed on the lifeline.
- Used to indicate that an object is active (or instantiated) during an interaction between two objects.
- The length of the rectangle indicates the duration of the objects staying active.

Interaction between two objects

- Occurs when one object sends a message to another.
- The use of the activation bar on the lifelines of the
  - Message Caller (the object that sends the message) and
  - Message Receiver (the object that receives the message)
  - Indicates that both are active/is instantiated during the exchange of the message

Activation bar example



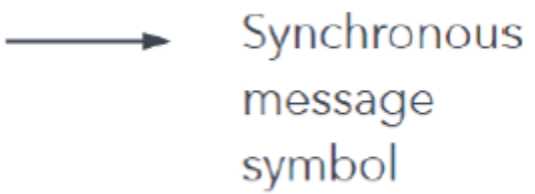
Basic Symbols and Components

Message Arrows

- An arrow
- from the Message Caller to the Message Receiver.
- A message can flow in any direction;
  - from left to right,
  - right to left or
  - back to the Message Caller itself.
- Different arrowheads
  - can indicate the type of message being sent or received.
- A message signature: message description
- *message_name (arguments): return_type*

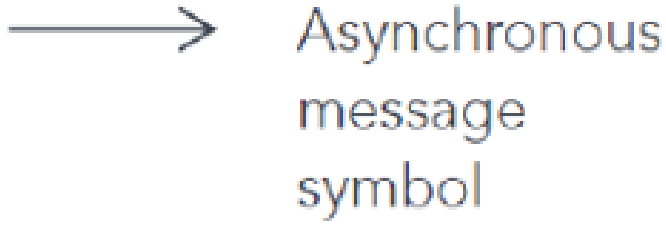
Common message symbols

- Represented by a **solid line with a solid arrowhead.**
- Sender waits for the receiver to process the message and return before carrying on with another message
- The diagram should show both the call and the reply.

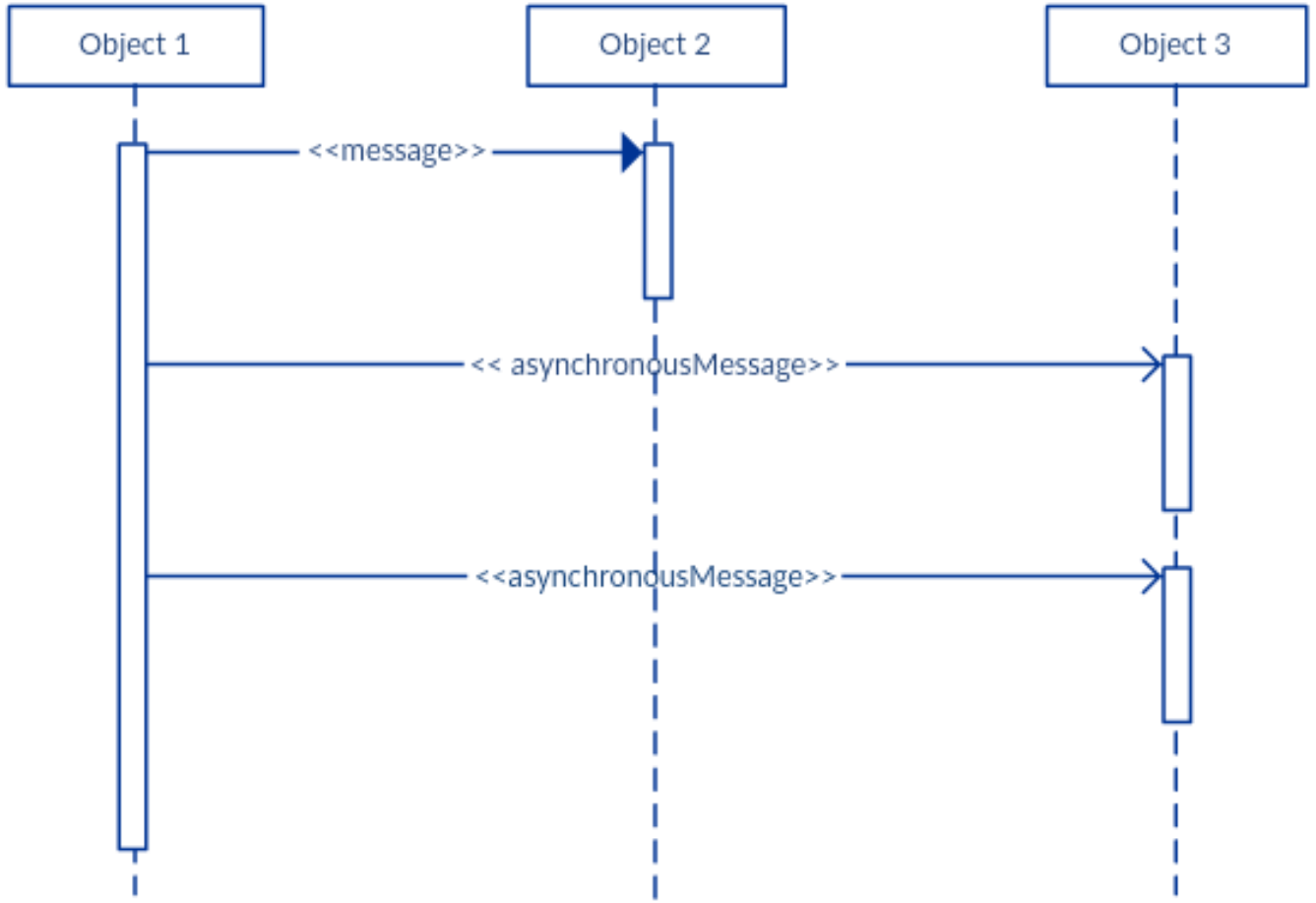


Common message symbols

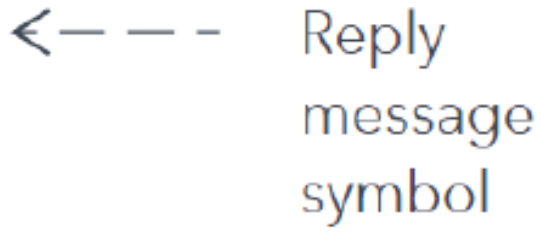
- Represented by **a solid line with a lined arrowhead.**
- Asynchronous messages **don't require a response before the sender continues.**
- Only the call should be included in the diagram



Example

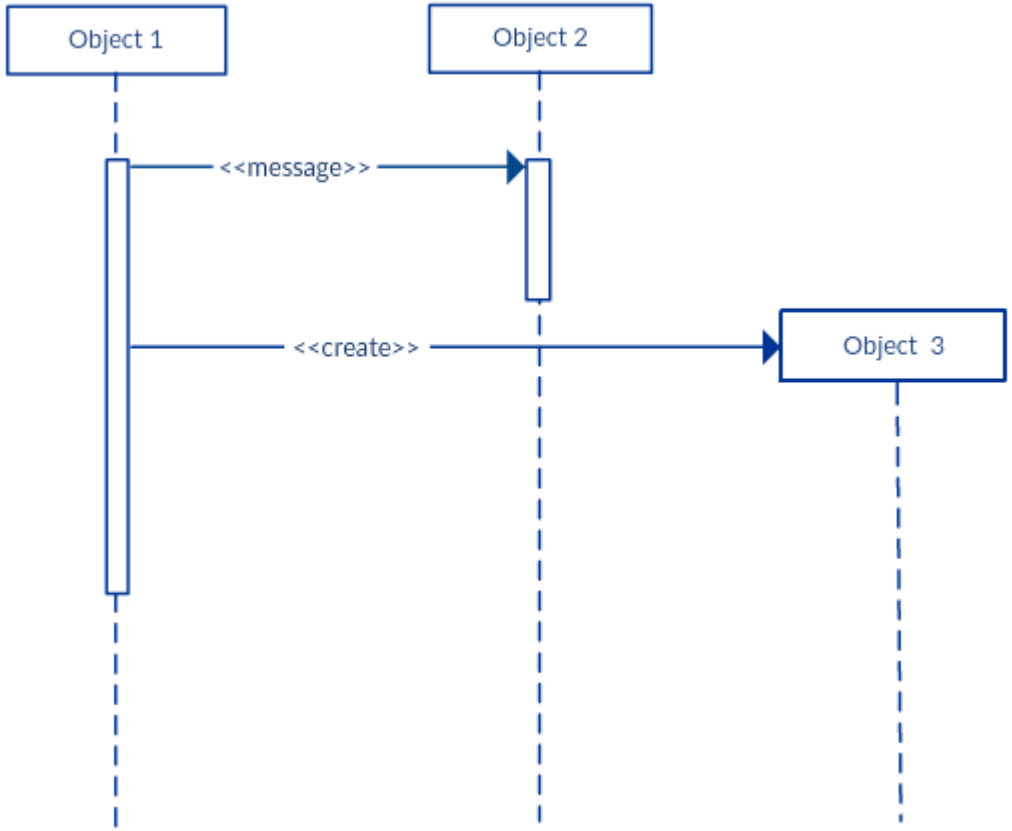

Object 1
Object 2
Object 3
<<message>>
<< asynchronousMessage>>
<<asynchronousMessage>>

Common message symbols



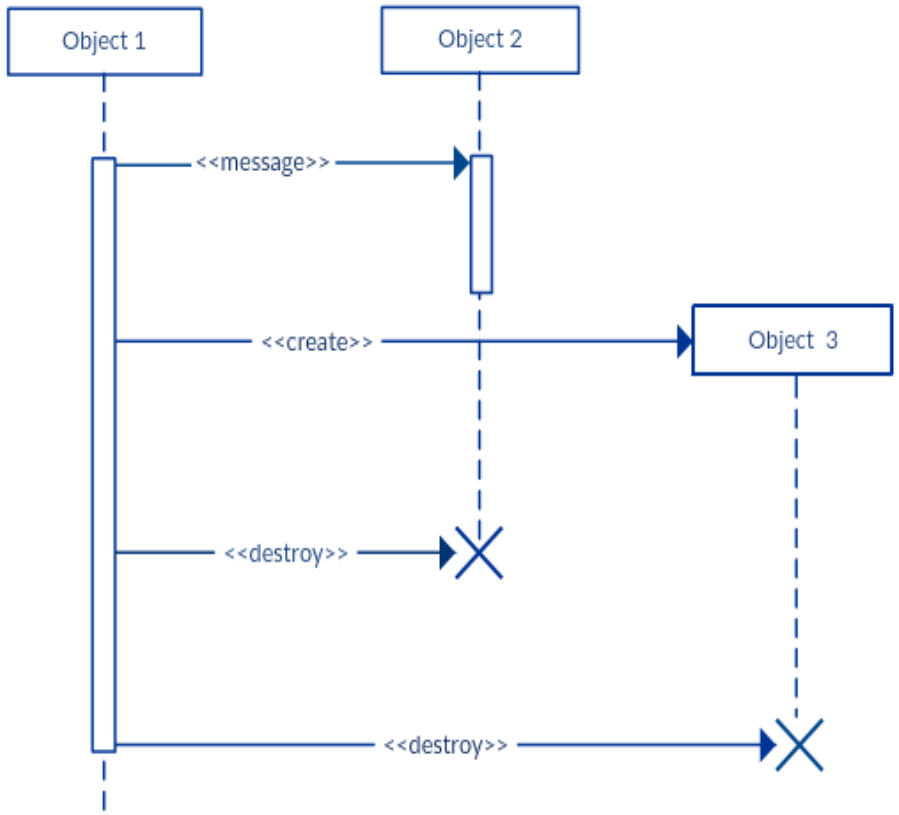
- Represented by a dashed line with a lined arrowhead.
- Indicate that the message receiver is done processing the message and is returning control over to the message caller

Participant Creation Message



- Objects do not necessarily live for the entire duration of the sequence of events.
- Objects or participants can be created according to the message that is being sent.

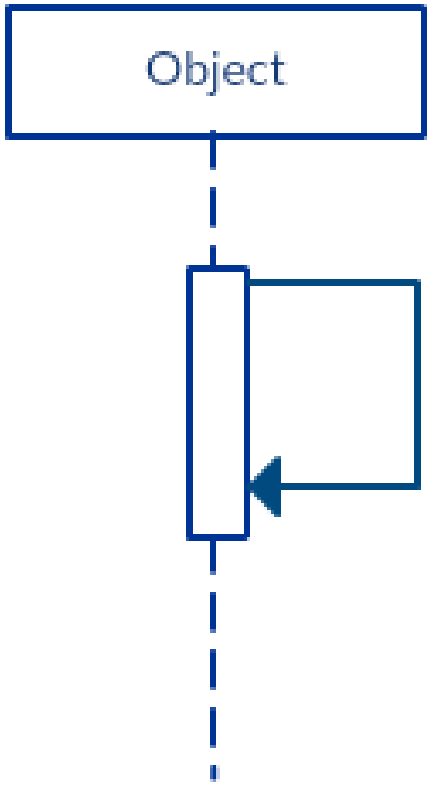
Participant destruction message



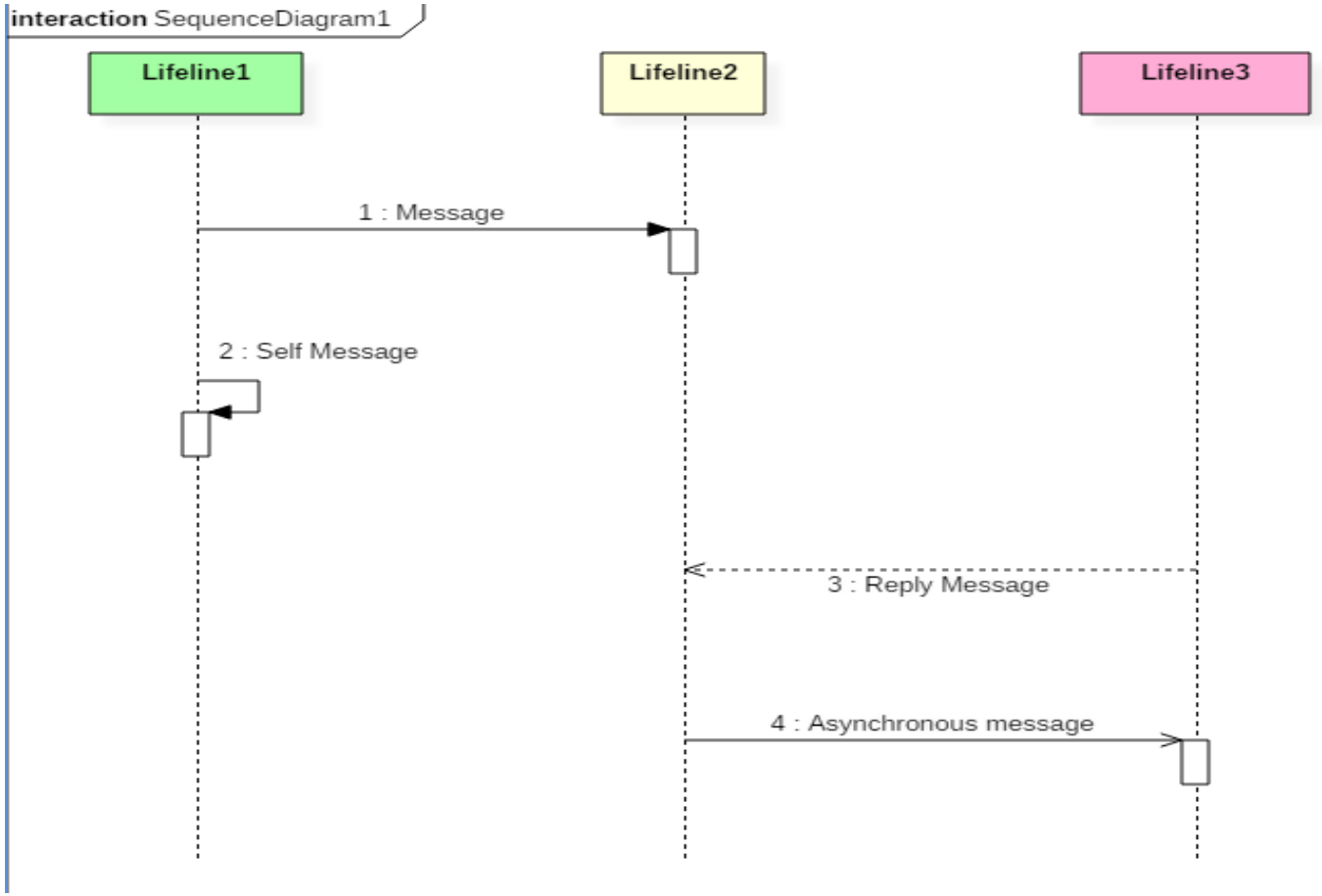
- When no longer needed participants can also be deleted from a sequence diagram.
- This is done by adding an ‘X’ at the end of the lifeline of the said participant.

Reflexive message

- When an object sends a message to itself, it is called a reflexive message.
- Indicated with a message arrow that starts and ends at the same lifeline.

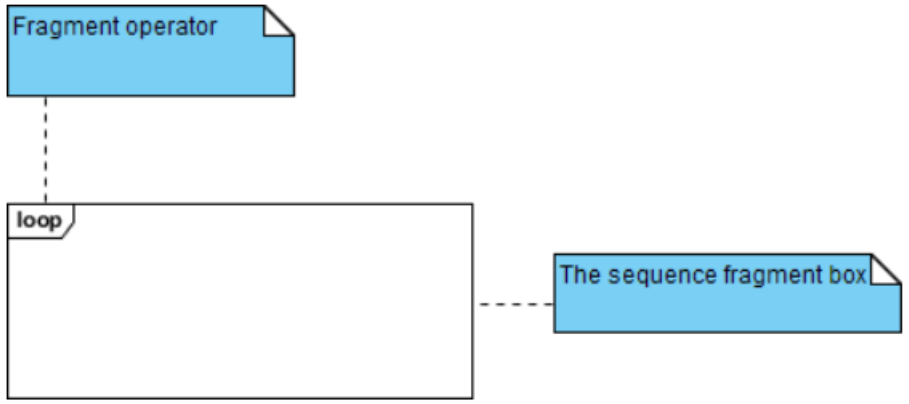


Sequence Modeling



Sequence Fragments

- UML 2.0 introduces sequence (or interaction) fragments.
- Sequence fragments make it easier to create and maintain accurate sequence diagrams
- A sequence fragment is represented as a box, called a combined fragment, which encloses a portion of the interactions within a sequence diagram
- The fragment operator (in the top left corner) indicates the type of fragment
- Fragment types: ref, assert, loop, break, alt, opt, neg



Sequence Fragments

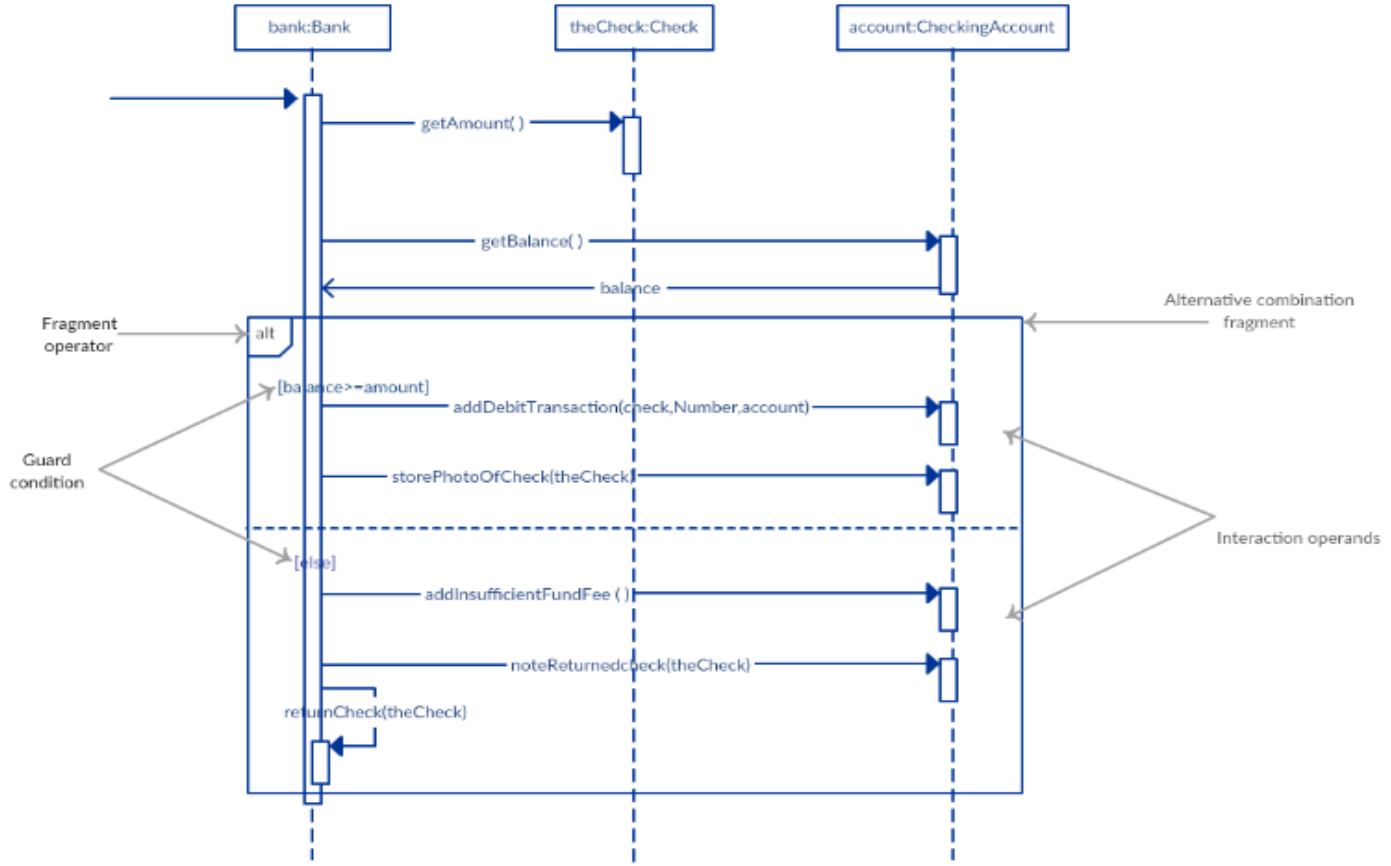
Operator	Fragment Type
alt	Alternative multiple fragments: only the one whose condition is true will execute.
opt	Optional: the fragment executes only if the supplied condition is true. Equivalent to an alt only with one trace.
par	Parallel: each fragment is run in parallel.
loop	Loop: the fragment may execute multiple times, and the guard indicates the basis of iteration.
region	Critical region: the fragment can have only one thread executing it at once.
neg	Negative: the fragment shows an invalid interaction.
ref	Reference: refers to an interaction defined on another diagram. The frame is drawn to cover the lifelines involved in the interaction. You can define parameters and a return value.

Sequence Fragments

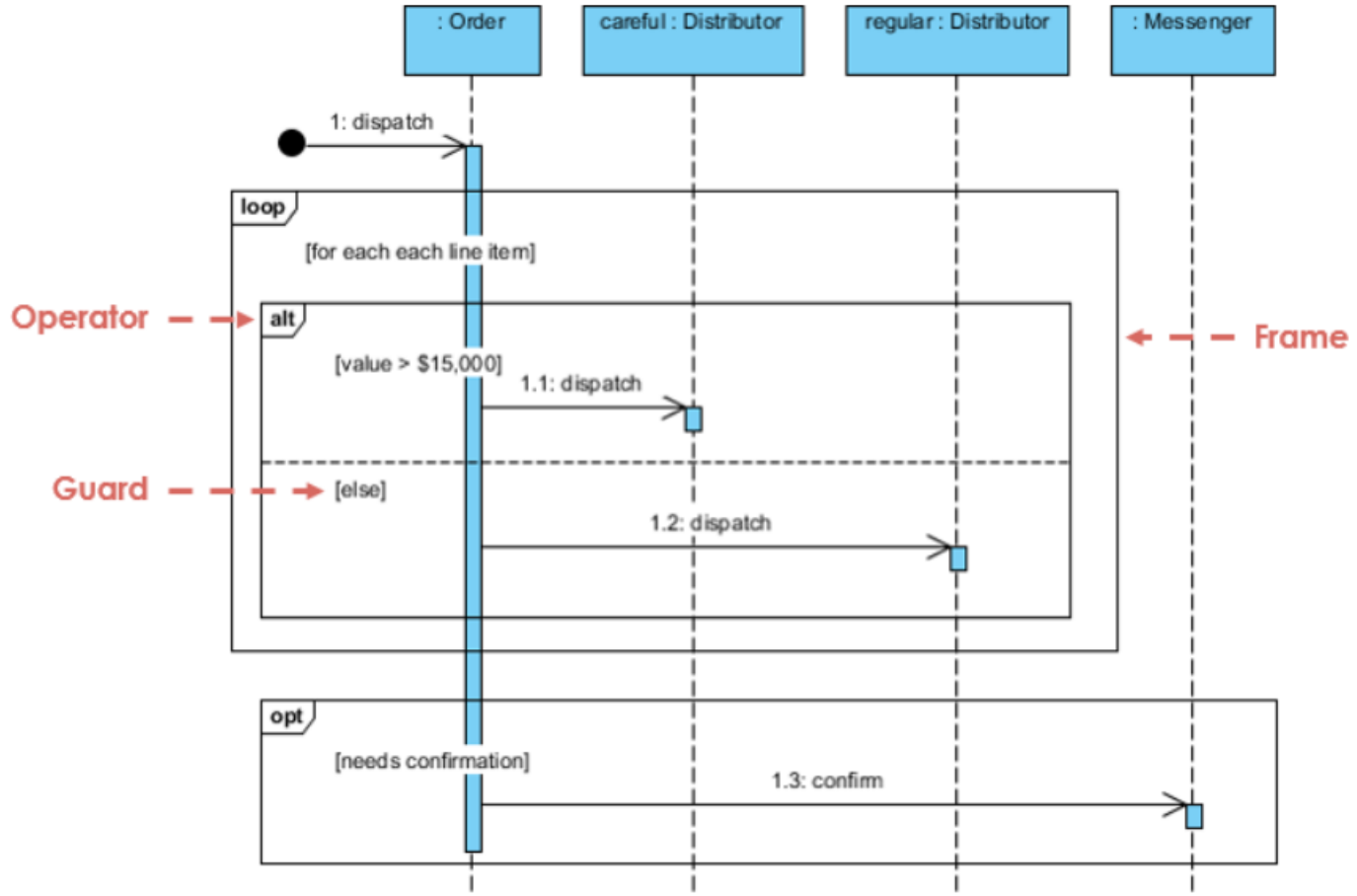
- To represent alternatives, use the labeled rectangle shape with a dashed line inside.

Alternative symbol

Alternative Symbol Example

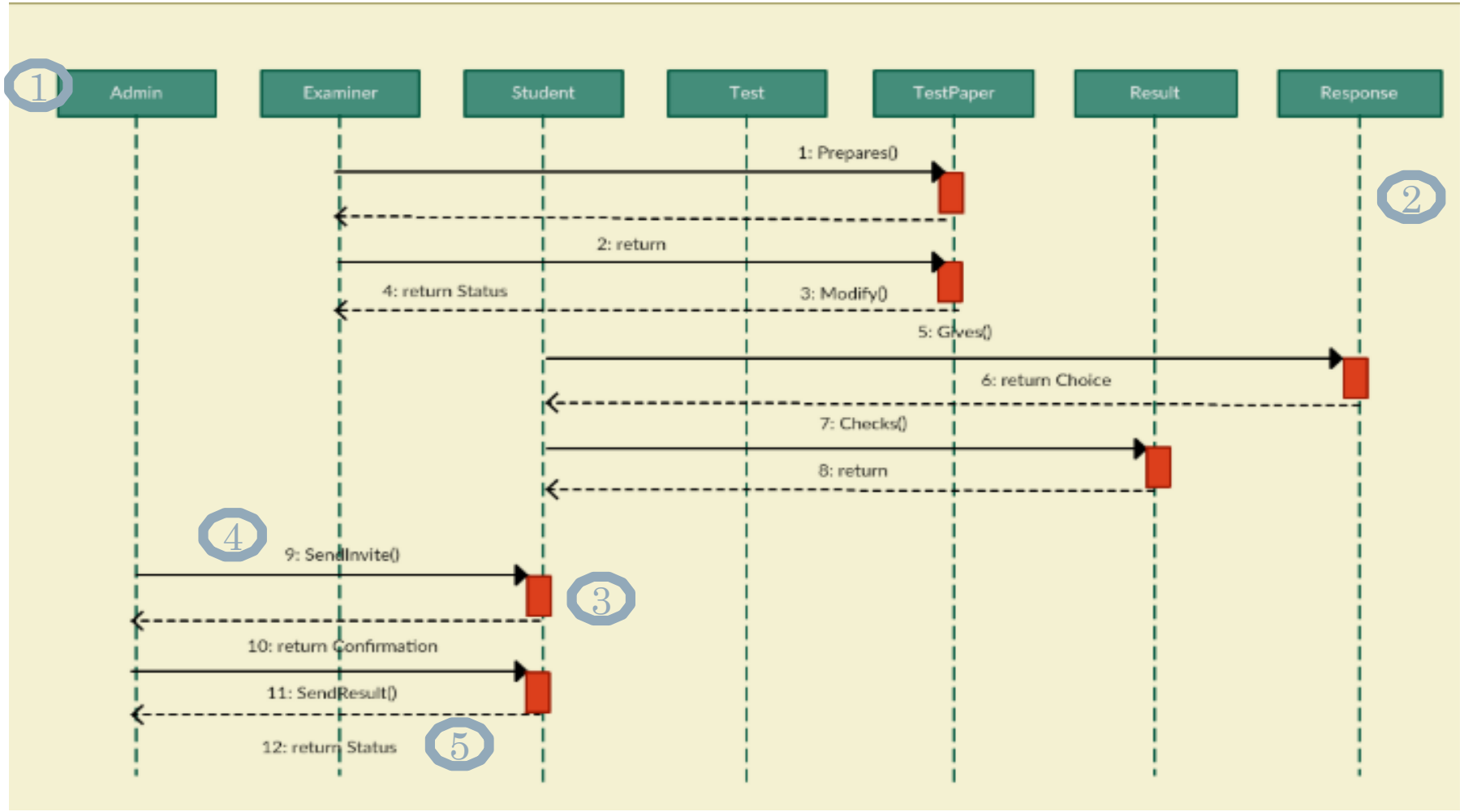


Combined Fragment Example



Activity

Identify the Elements



Consider the following use case narration describing the scenario of a specially designed telephone box used to take calls using a credit card.

Draw a sequence diagram to represent the scenario.

Use Case Name :	**Take a telephone call using the credit card**
Participant:	**User** **Credit card processing center**
Entry Conditions:	**The user has a valid credit card** **The credit balance should be sufficient for the minimum amount required to take a phone call.**
Exit Conditions:	**User hangs the receiver** **Credit card is returned to the user**

Flow of Events:

1. The user inserts the credit card.

2. Telephone box sends the message to the card center.

3. Card center processes the card.

4. Card center sends the available credit to the Telephone box.

5. Telephone box processes the available balance.

6. Telephone box shows a message for the user to pick up the receiver.

7. User picks up the receiver.

8. Telephone box displays the number pad to the user on the screen.

9. User dials the number and makes the phone call.

10. Telephone box processes the duration/amount.

11. User ends the phone call by pressing the 'END' button / hanging the receiver.

12. Telephone box displays the total phone call duration and the amount.

13. Telephone box sends the amount to the card center.

14. Card center charges the amount from the credit card.

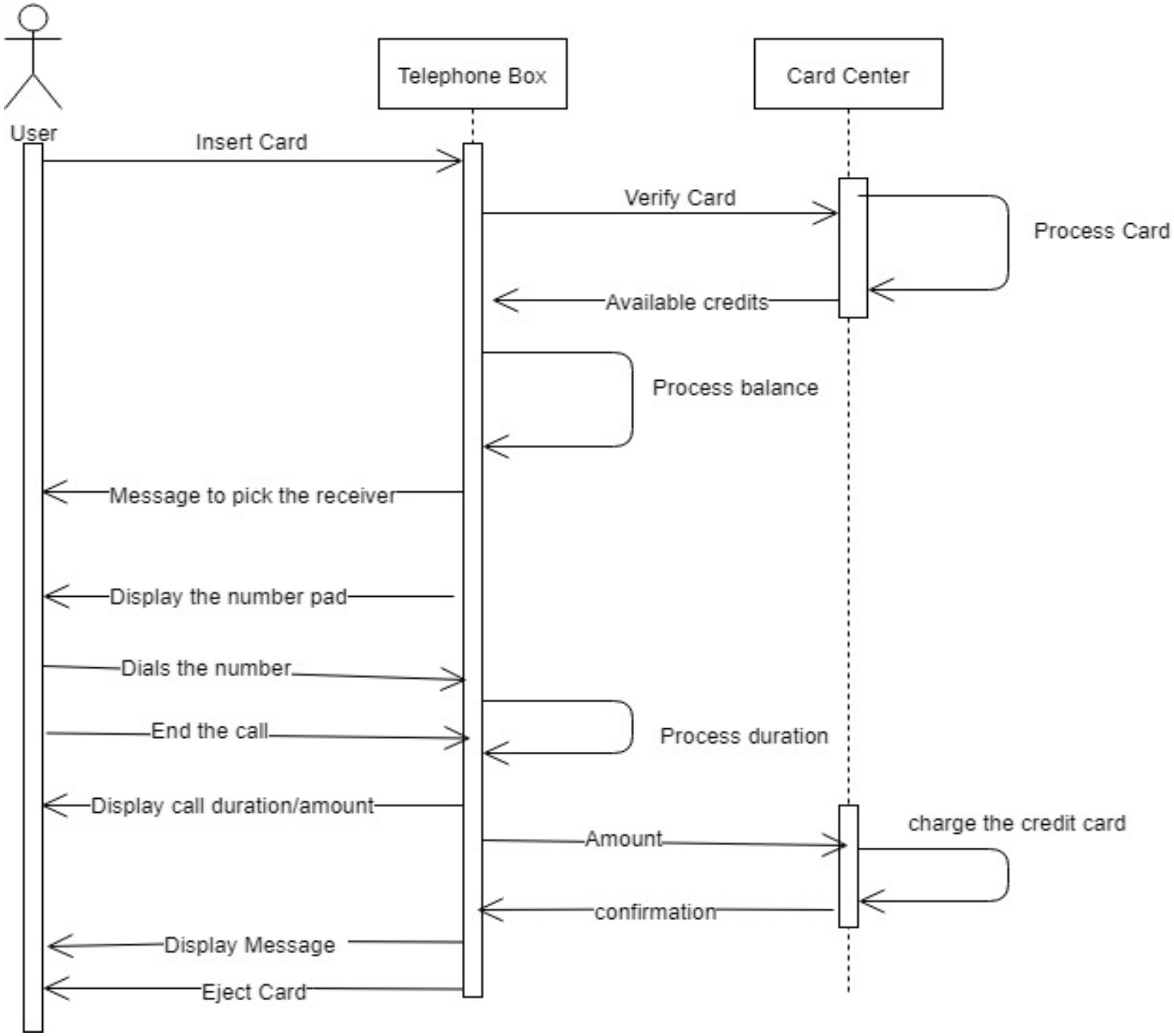
15. Card center sends the confirmation to the telephone box.

16. Telephone box displays the message 'THANK YOU, COME AGAIN".

17. Telephone box ejects the credit card.

Exceptional conditions and alternative flow of events:

When the credit is not sufficient to make a call: Display error message and Go to 17
When the amount exceeds the credit card balance: Ends the phone call and Go to 12


User
Telephone Box
Card Center
Insert Card
Verify Card
Process Card
Available credits
Process balance
Message to pick the receiver
Display the number pad
Dials the number
End the call
Process duration
Display call duration/amount
Amount
charge the credit card
confirmation
Display Message
Eject Card

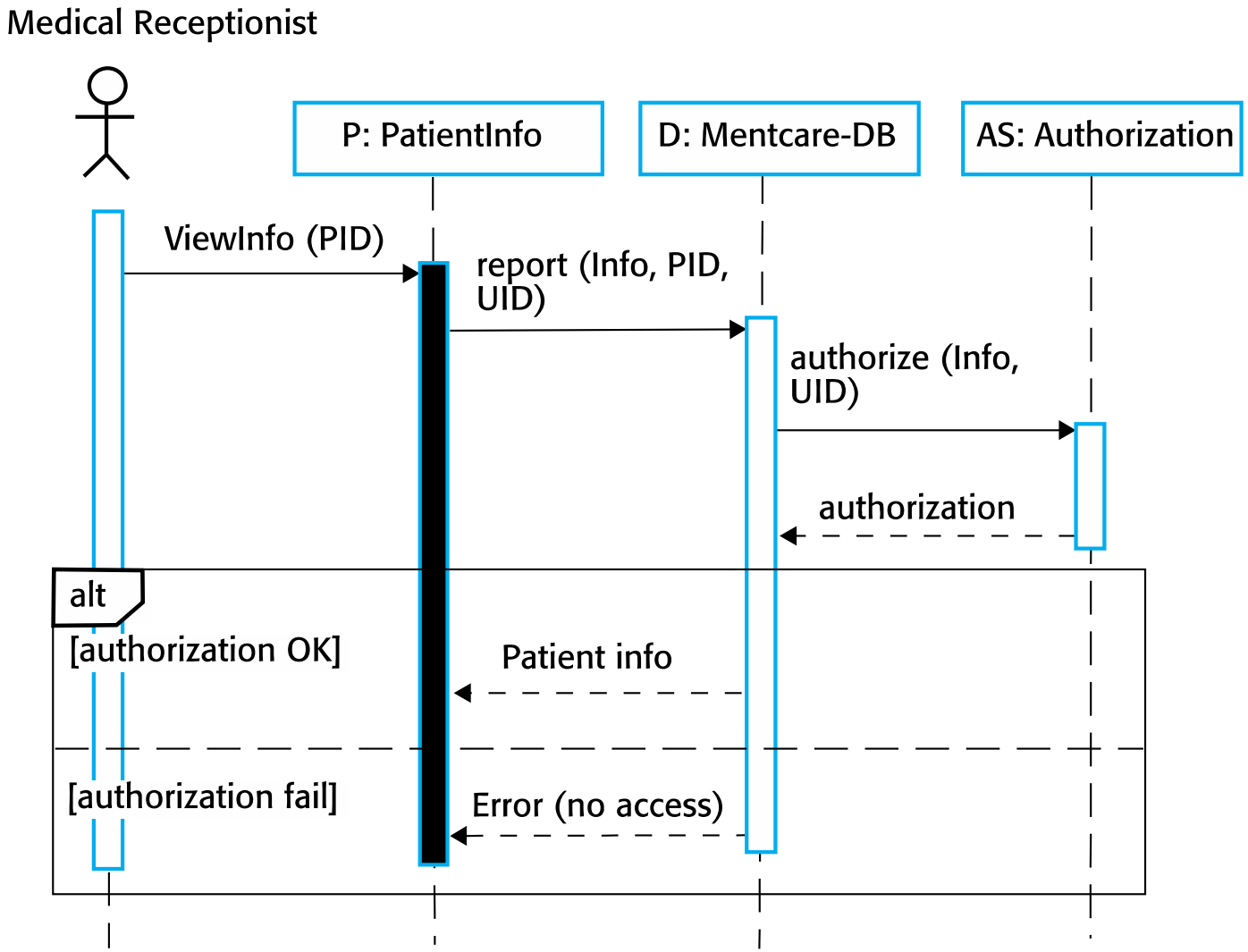
Use-Case Narrative

Author (s): ---------- **Date:----------**
Version: ----------

Use-Case Name:		**Use-Case Type** **Business Requirements:** ☐
Use-Case ID:		
Priority:		
Source:		
Primary Business Actor:		
Other Participating Actors:		
Other Interested Stakeholders:		
Description:		

There is no standard template for Use Case Narratives.

Describe - View Patient Info



Describe - View Patient Info

- The medical receptionist triggers the ViewInfo method in **an instance P of the PatientInfo object class**, supplying the patient's identifier, PID to identify the required information. P is a user interface object that is displayed as a form showing patient information.
- The instance P calls the database to return the information required, supplying the receptionist's identifier to allow security checking.
- The database checks with an authorization system that the receptionist has authorized for this action.
- If authorized, the patient information is returned and is displayed on a form on the user's screen.
- If authorization fails, then an error message is returned. The box denoted by "alt" in the top-left corner is a choice box indicating that one of the contained interactions will be executed. The condition that selects the choice is shown in square brackets.

A sequence diagram represents the scenario or flow of events in one single use case.

The message flow of the sequence diagram is based on the narrative of the particular use case.

Communication Diagram

Interaction Models

Communication Diagrams

- Provide a view of the dynamic aspects of an object-oriented system.
- Shows how the members of a set of objects collaborate to implement a use case or a use-case scenario.
- Emphasize the flow of messages through a set of objects, whereas the sequence diagrams focus on the time ordering of the messages.
- Used to model the flow of control over a set of collaborating objects or to model which objects collaborate to support business processes.

Elements of a Communication Diagram

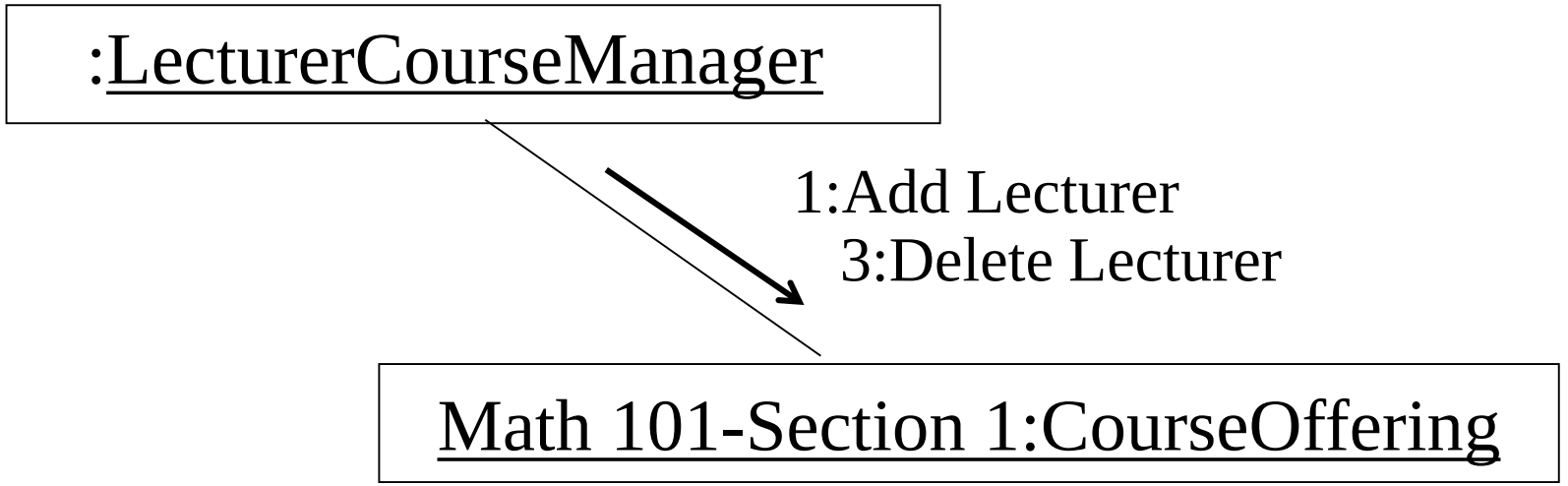
Term and Definition	Symbol
An actor: ■ Is a person or system that derives benefit from and is external to the system. ■ Participates in a collaboration by sending and/or receiving messages. ■ Is depicted either as a stick figure (default) or, if a nonhuman actor is involved, as a rectangle with <<actor>> in it (alternative).	 anActor <<actor>> anActor
An object: ■ Participates in a collaboration by sending and/or receiving messages.	anObject : aClass
An association: ■ Shows an association between actors and/or objects. ■ Is used to send messages.	
A message: ■ Conveys information from one object to another one. ■ Has direction shown using an arrowhead. ■ Has sequence shown by a sequence number.	SeqNumber: aMessage →
A guard condition: ■ Represents a test that must be met for the message to be sent.	SeqNumber: [aGuardCondition]: aMessage →
A frame: ■ Indicates the context of the communication diagram.	Context

Creating a Communication Diagram

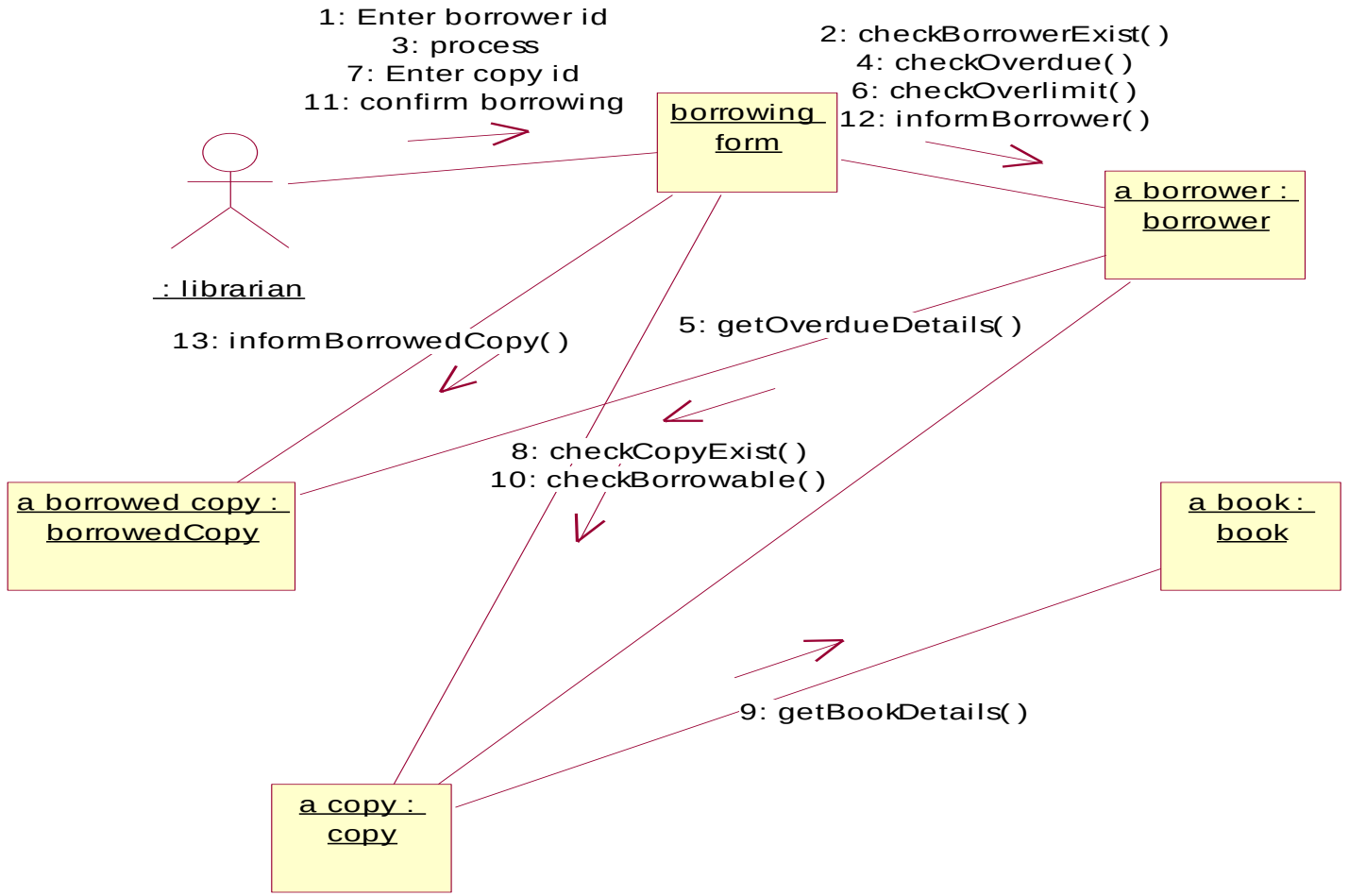
- An alternative way to show a scenario.
- Shows object interactions organized around the objects and their links to each other.
- A communication diagram contains:
  - Objects drawn as rectangles.
  - Links between objects are shown as lines connecting the linked objects.
  - Messages are shown as text and an arrow that points from the client to the supplier.

Communication Diagrams

UML Notation for objects, links and messages in a communication diagram.



Communication Diagrams



Sequence vs. Communication Diagrams

Feature	Sequence Diagram	Communication Diagram
Focus	Time sequence of interactions	Structural relationships between objects
Representation	Emphasizes order of messages	Shows how objects are linked
Use Case	Best for understanding execution flow	Best for analyzing system structure
Message Numbering	Implicit (top to bottom)	Explicit numbering required
Complexity Handling	Suitable for complex scenarios with many steps	Good for understanding system architecture