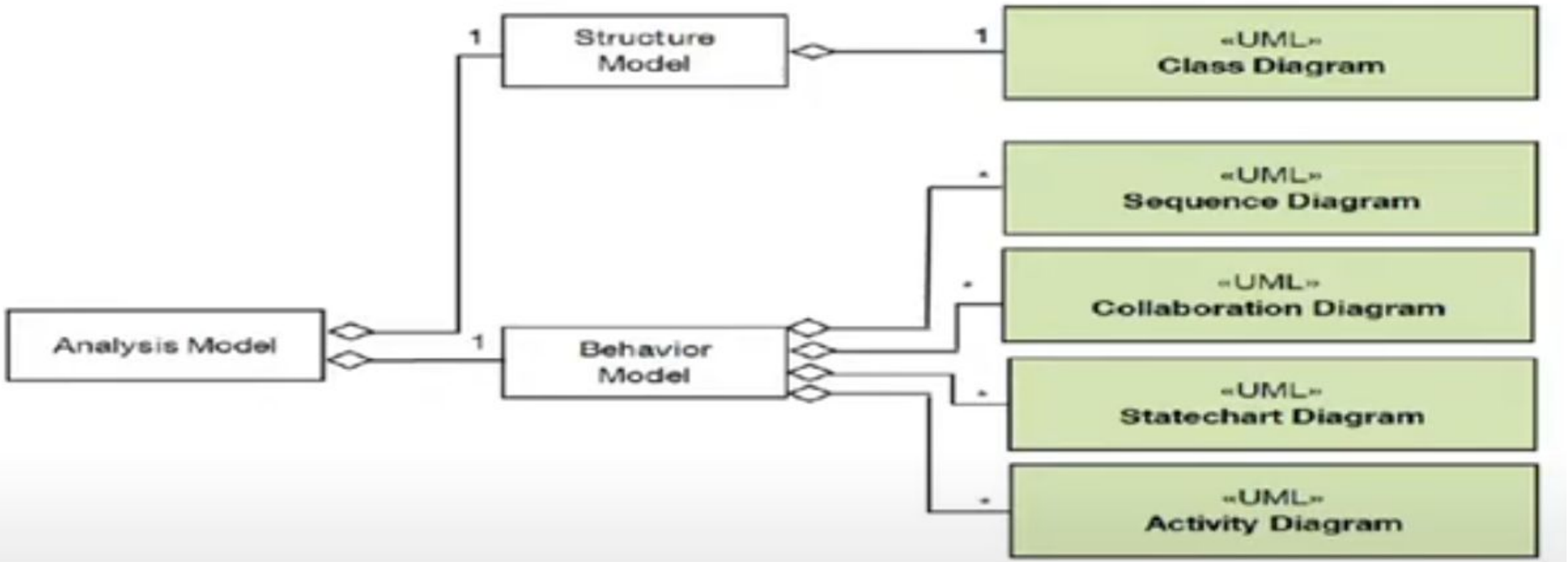
Object-Oriented Analysis and Design using JAVA

B.Tech (CSE/IT) 5th SEM
2020-2021

Lecture-20 Collaboration diagram or Communication diagram

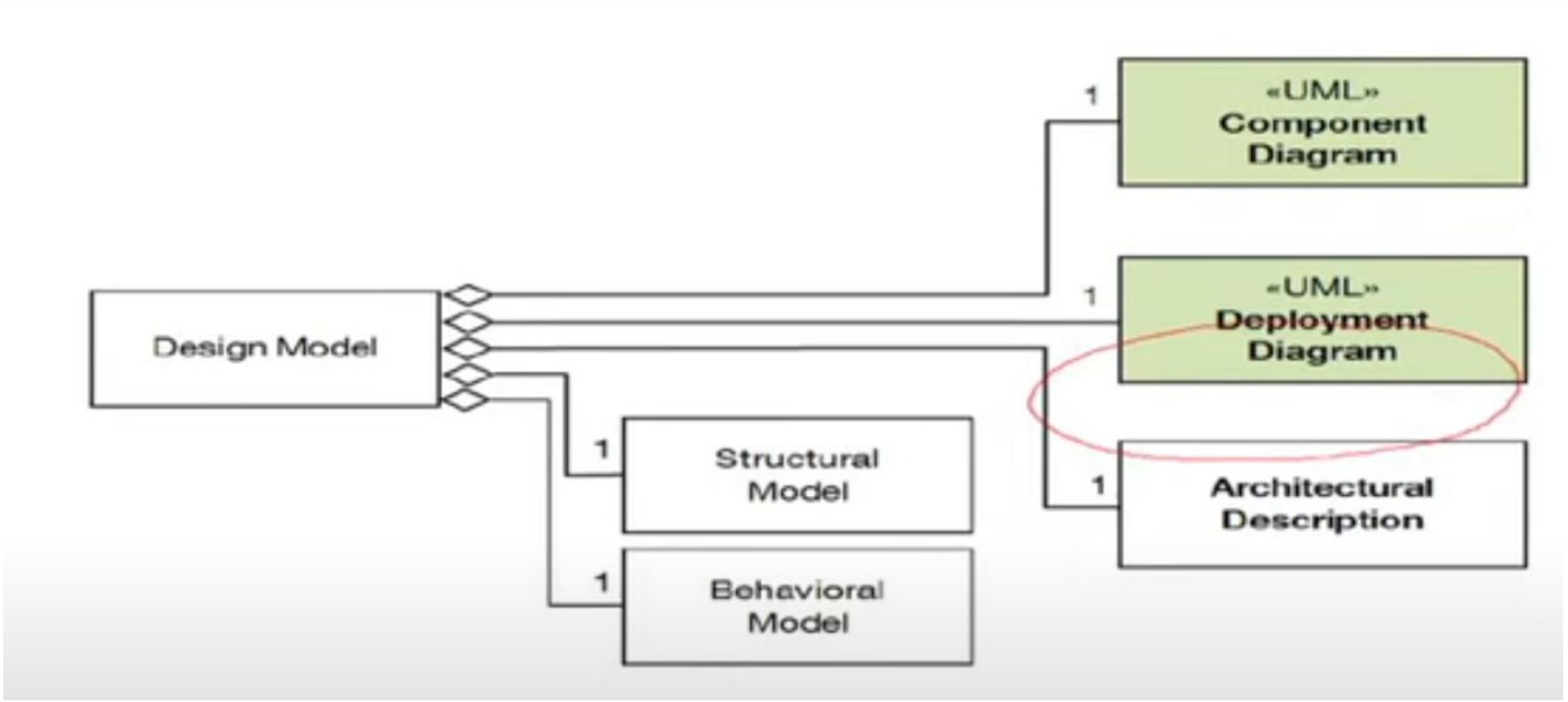
Where used?

Communication diagram is a result of analysis phase



Where used?

Refined in the design phase and included in behavioral model



Introduction

- The collaboration diagram is used to show the relationship between the objects in a system.
- Both the sequence and the collaboration diagrams represent the same information but differently.
- Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.
- An object consists of several features. Multiple objects present in the system are connected to each other.
- The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

Reference: https://www.javatpoint.com/uml-collaboration-diagram/

Notations of a Collaboration Diagram

Following are the components of a component diagram that are enlisted below:

1. **Objects:** The representation of an object is done by an object symbol with its name and class underlined, separated by a colon.

In the collaboration diagram, objects are utilized in the following ways:

1. The object is represented by specifying their name and class.
2. It is not mandatory for every class to appear.
3. A class may constitute more than one object.
4. In the collaboration diagram, firstly, the object is created, and then its class is specified.
5. To differentiate one object from another object, it is necessary to name them.

Reference: https://www.javatpoint.com/uml-collaboration-diagram/

•**Actors:** In the collaboration diagram, the actor plays the main role as it invokes the interaction. Each actor has its respective role and name. In this, one actor initiates the use case.

•**Links:** The link is an instance of association, which associates the objects and actors. It portrays a relationship between the objects through which the messages are sent. It is represented by a solid line. The link helps an object to connect with or navigate to another object, such that the message flows are attached to links.

Reference: https://www.javatpoint.com/uml-collaboration-diagram/

Steps for Creating Collaboration Diagrams

1. Identify behavior whose realization and implementation is specified

2. Identify the structural elements (class roles, objects, subsystems) necessary to carry out the functionality of the collaboration
    1. Decide on the context of interaction: system, subsystem, use case and operation

3. Model structural relationships between those elements to produce a diagram showing the context of the interaction

4. Consider the alternative scenarios that may be required
    1. Draw instance level collaboration diagrams, if required.
    2. Optionally draw a specification level collaboration diagram to summarize the alternative scenarios in the instance level sequence diagrams

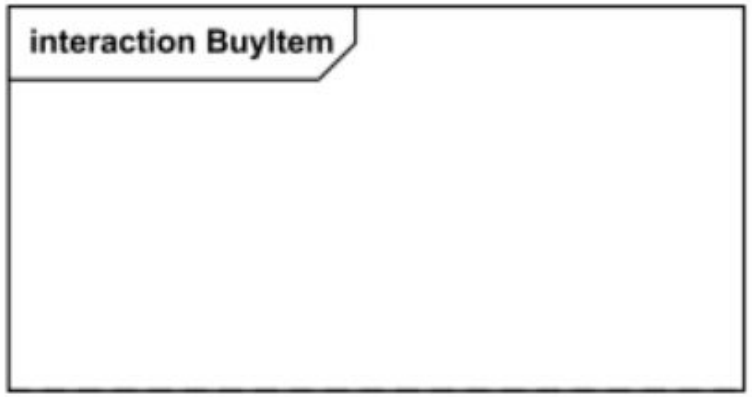
Reference: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml-collaboration-diagram/

Major components

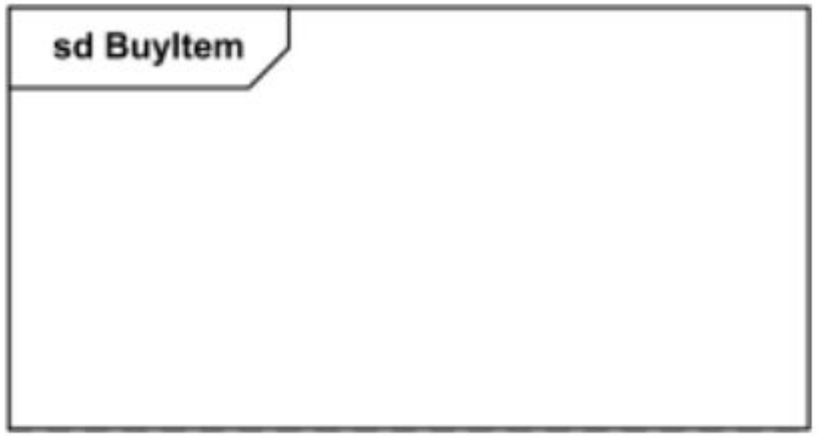
- Frames
- Lifeline
- Messages

Frame

- Communication diagrams could be shown within a rectangular frame with the name in a compartment in the upper left corner.



Interaction ***Frame*** *for* ***Communication Diagram*** *BuyItem*



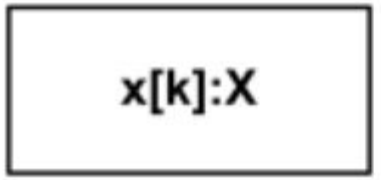
Sd ***Frame*** *for* ***Communication Diagram*** *BuyItem*

Lifeline

- Lifeline is a specialization of named element which represents an individual participant in the interaction.
- The lifeline head has a shape that is based on the classifier for the part that this lifeline represents. Usually the head is a white rectangle containing name of the class after colon.

Anonymous lifeline of class User.

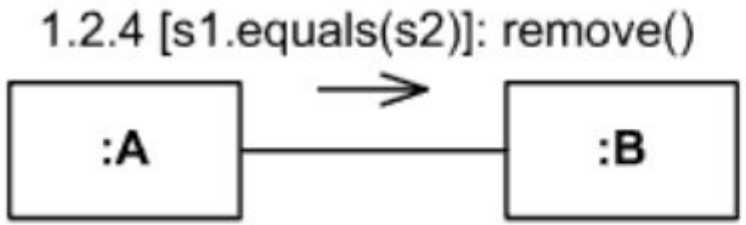
***Lifeline** "data" of class Stock*



***Lifeline** "x" of class X is selected with **selector** [k].*

Message

- Message in communication diagram is shown as a line with sequence expression and arrow above the line. The arrow indicates direction of the communication.



Instance of class A sends remove() message to instance of B if s1 is equal to s2

Sequence Expression

- The sequence expression is a dot separated list of sequence terms followed by a colon (":") and message name after that:

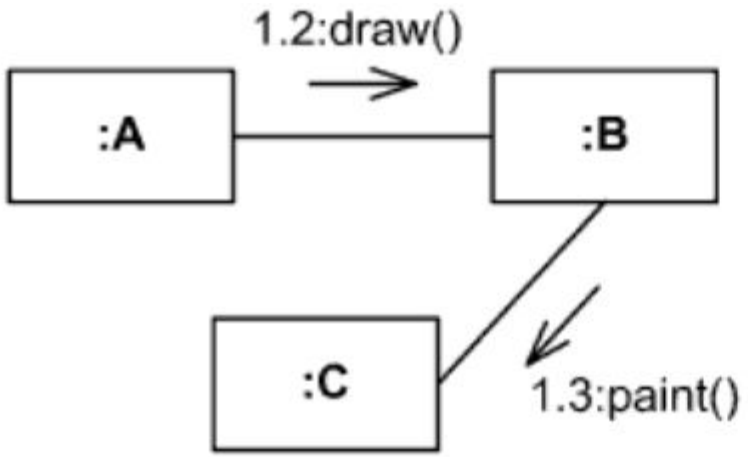
  sequence-expression ::= sequence-term '.' . . . ':' message-name
- For example, **3b.2.2:m5**
  - contains sequence expression 3b.2.2 and message name m5.

sequence-term ::= [***integer*** [***name***]] [***recurrence***]

The integer represents the sequential order of the message within the next higher level of procedural calling (activation). Messages that differ in one integer term are sequential at that level of nesting.

For example,

- message with sequence 2 follows message with sequence 1,
- 2.1 follows 2
- 5.3 follows 5.2 within activation 5
- 1.2.4 follows message 1.2.3 within activation 1.2.

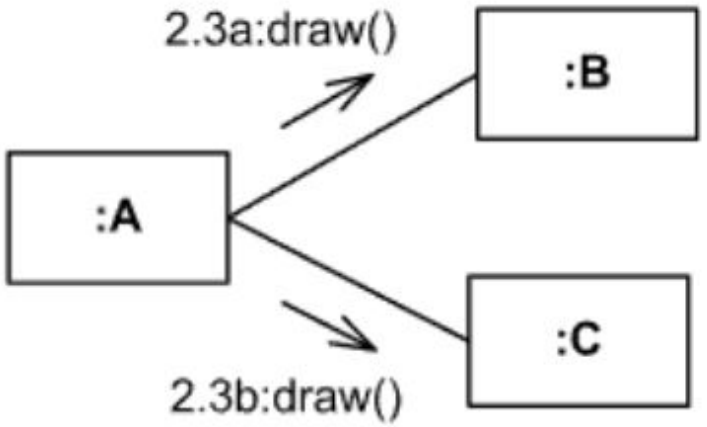


Instance of A sends draw() message to instance of B, and after that B sends paint() to C

The **name** represents a **concurrent thread** of control. Messages that differ in the final name are concurrent at that level of nesting.

For example,

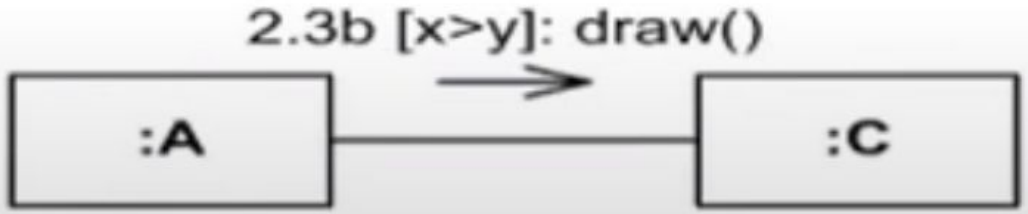
- messages 2.3a and 2.3b are concurrent within activation 2.3,
- 1.1 follows 1a and 1b,
- 3a.2.1 and 3b.2.1 follow 3.2.



Instance of A sends draw() messages concurrently to instance of B and to instance of C

Sequence Expression: Guards

- A **guard** specifies condition for the message to be sent (executed) at the given nesting depth
- **Example**:
  - **2.3b [x>y]: draw()**: message draw() will be executed if x is greater than y
  - **1.1.1 [s1.equals(s2)]: remove()** – message remove() will be executed if s1 equals s2



Instance of class A will send message draw() to the instance of C, if x > y

Sequence Expression: Recurrence and Iteration

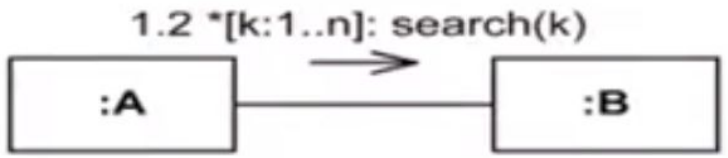
- The **recurrence** defines conditional or iterative execution of zero or more messages that are executed depending on the specified condition
  recurrence ::= branch | loop , branch ::= '[' guard ']'
- An **iteration** specifies a sequence of messages at the given nesting depth
- **Notation**:
  - * : *Messages Executed Sequentially*
  - *|| : *Messages Executed Concurrently*

Example:

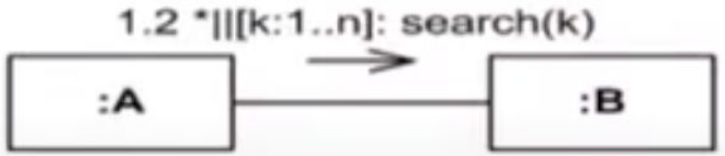
- **4.2c *[i=1..12]: search(t[i])** – search() will be executed 12 times, one after another
- **4.2c *||[i=1..12]: search(t[i])** – 12 search() messages will be sent concurrently
- **2.2 *: notify()** – message notify() will be repeated some unspecified number of times

Sequence Expression: Recurrence and Iteration

Example:

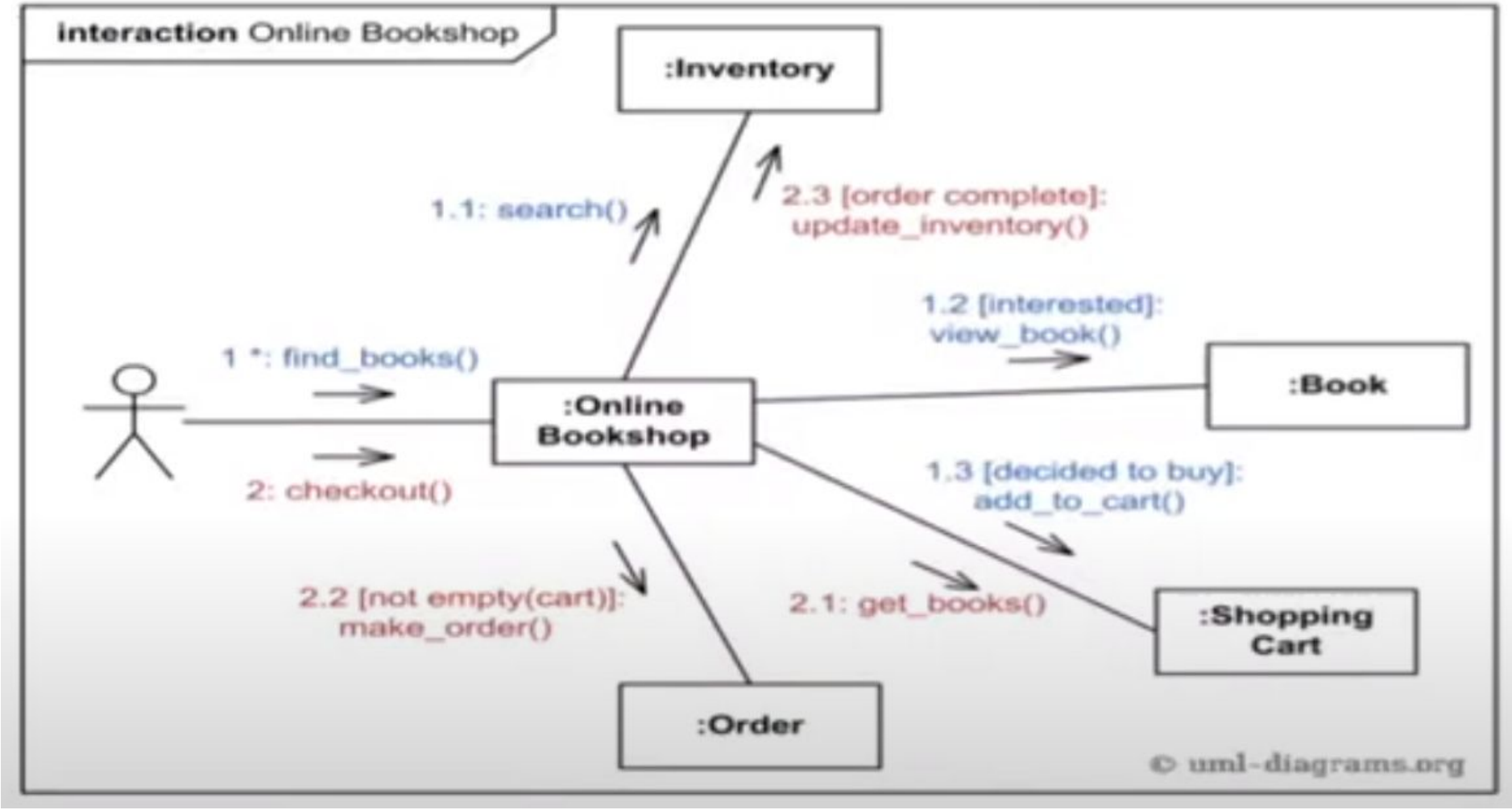


Instance of class A will send search() message to instance of B n times, one by one



Instance of class A will send n concurrent search() messages to instance of B

Online Book Shop



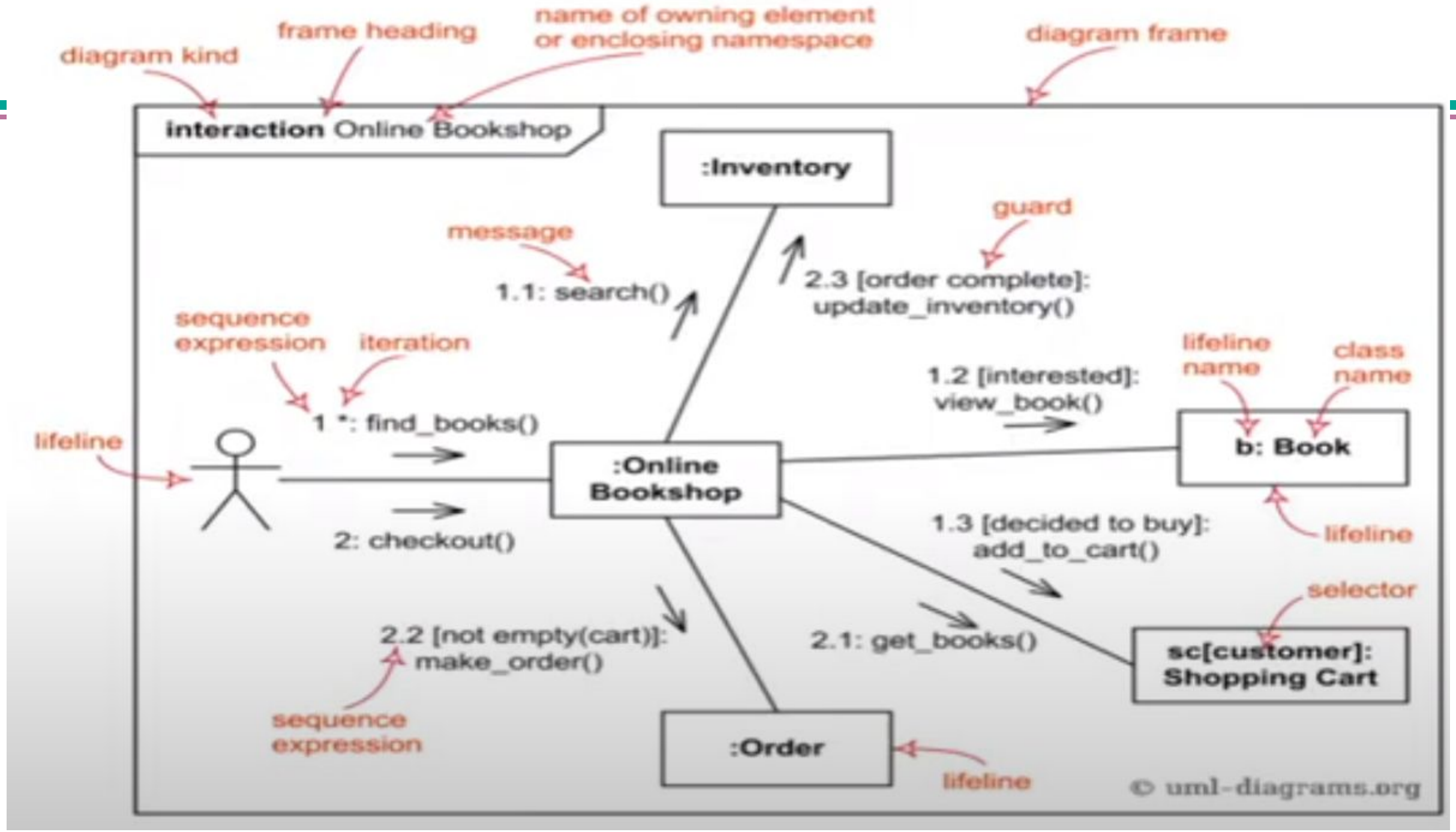
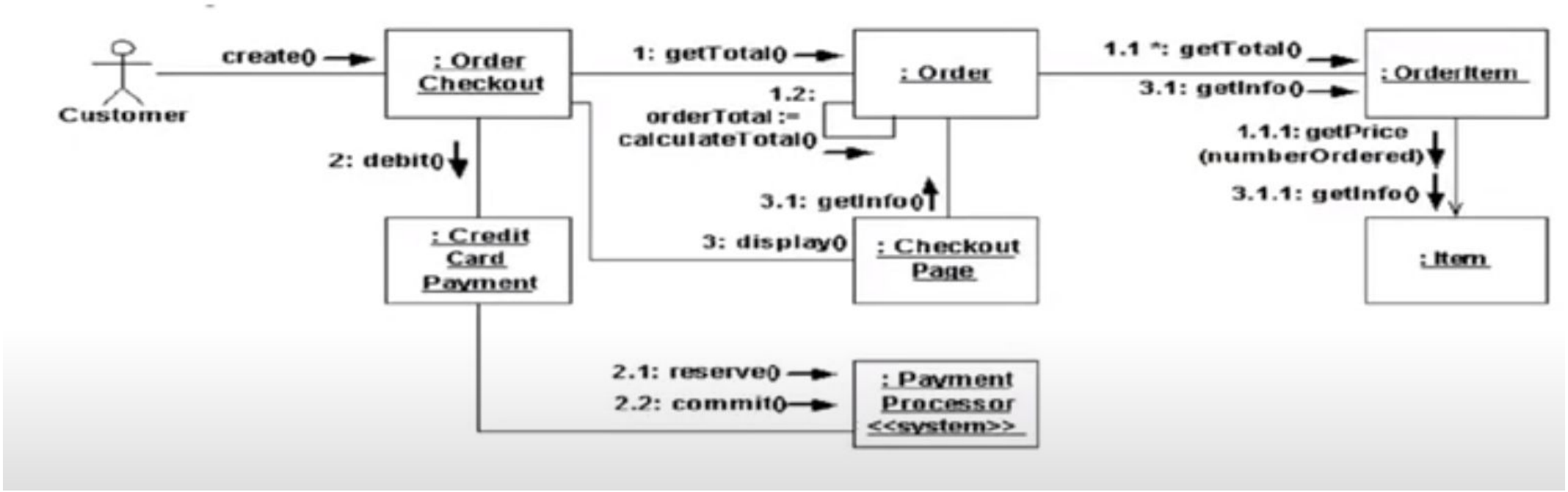
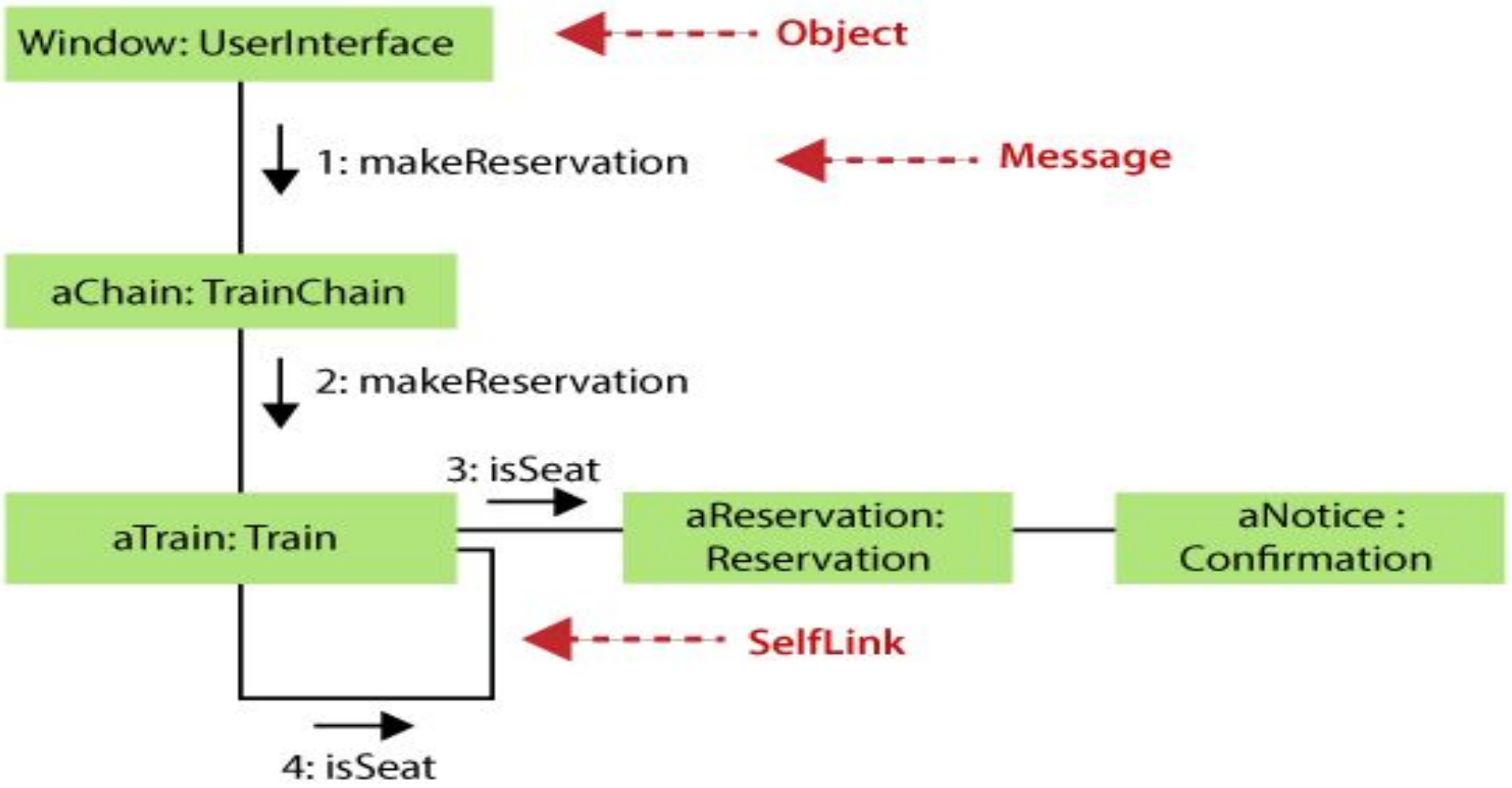


diagram kind
frame heading
name of owning element
or enclosing namespace
diagram frame
interaction Online Bookshop
:Inventory
guard
message
1.1: search()
2.3 [order complete]:
update_inventory()
sequence
expression
iteration
1 *: find_books()
1.2 [interested]:
view_book()
lifeline
name
class
name
lifeline
:Online
Bookshop
b: Book
2: checkout()
1.3 [decided to buy]:
add_to_cart()
lifeline
selector
2.2 [not empty(cart)]:
make_order()
2.1: get_books()
sc[customer]:
Shopping Cart
sequence
expression
:Order
lifeline
© uml-diagrams.org

Order Management

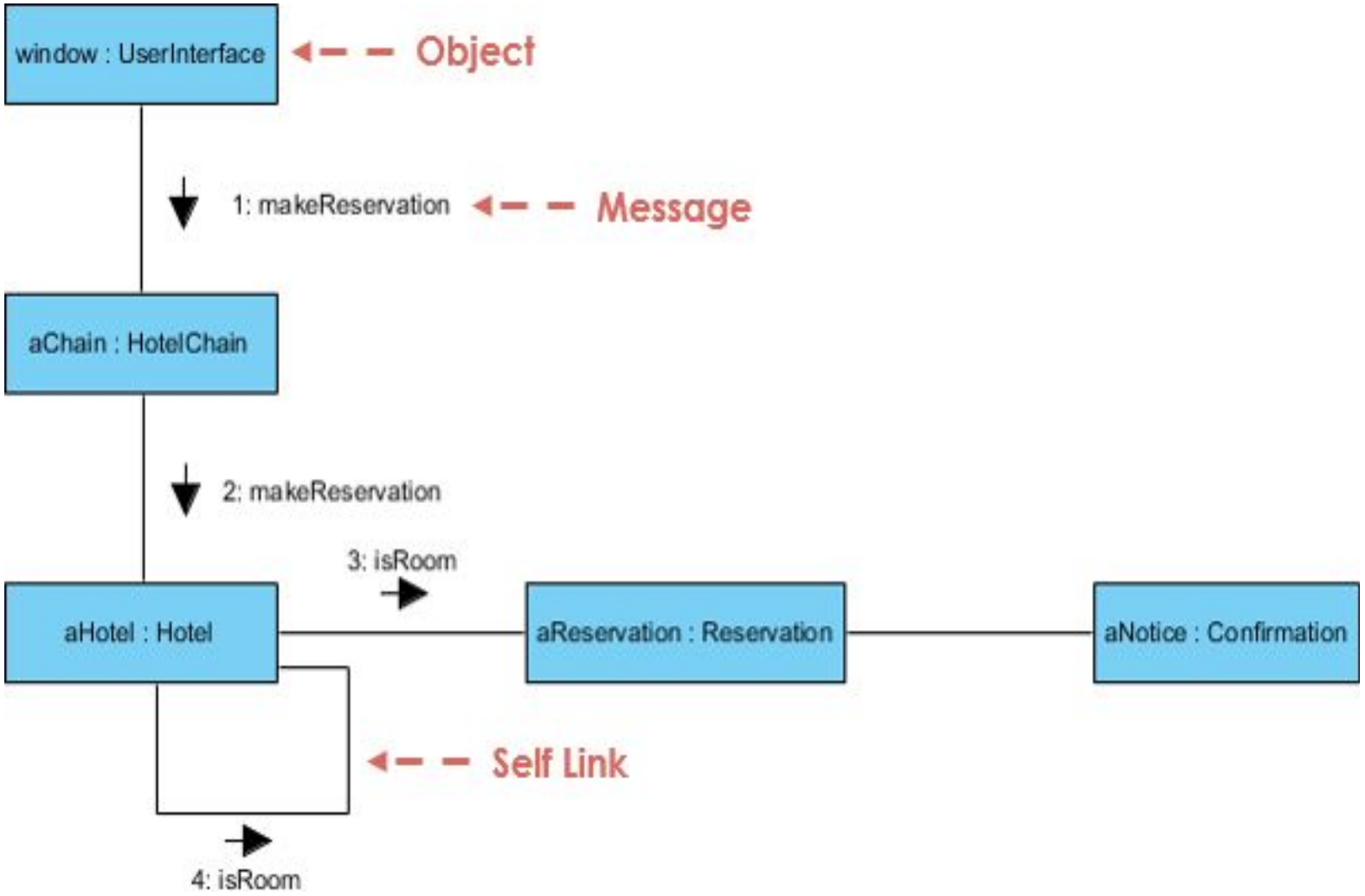


Collaboration Diagram Example



Reference: https://www.javatpoint.com/uml-collaboration-diagram/

Collaboration Diagram Example



Reference: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml-collaboration-diagram/

Benefits of Collaboration Diagram

- The collaboration diagram is also known as Communication Diagram. It mainly puts emphasis on the structural aspect of an interaction diagram, i.e., how lifelines are connected.
- The syntax of a collaboration diagram is similar to the sequence diagram; just the difference is that the lifeline does not consist of tails.
- The messages transmitted over sequencing is represented by numbering each individual message. The collaboration diagram is semantically weak in comparison to the sequence diagram.
- The special case of a collaboration diagram is the object diagram. It focuses on the elements and not the message flow, like sequence diagrams.
- Since the collaboration diagrams are not that expensive, the sequence diagram can be directly converted to the collaboration diagram.
- There may be a chance of losing some amount of information while implementing a collaboration diagram with respect to the sequence diagram.

Reference: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml-collaboration-diagram/

Key references

Unified Modeling Language (UML): Complete Guide & Examples
-By James Rumbaugh, Ivar Jacobson, Grady Booch

https://www.javatpoint.com/uml-collaboration-diagram/ :
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml-collaboration-diagram/

Key references

Unified Modeling Language (UML): Complete Guide & Examples
-By James Rumbaugh, Ivar Jacobson, Grady Booch

https://www.guru99.com/uml-activity-diagram.html
https://creately.com/blog/diagrams/sequence-diagram-tutorial/

Thank You