

Collaboration Diagram

Lecture # 31,32
16,18 Nov

Rubab Jaffar
rubab.jaffar@nu.edu.pk

Software Design and Analysis CS-324



Today's Outline

- Collaboration Diagram Semantics
- Collaboration Diagram Notation
- Collaboration Diagram Examples
- Collaboration Diagram Issues

Collaboration Diagram Semantics

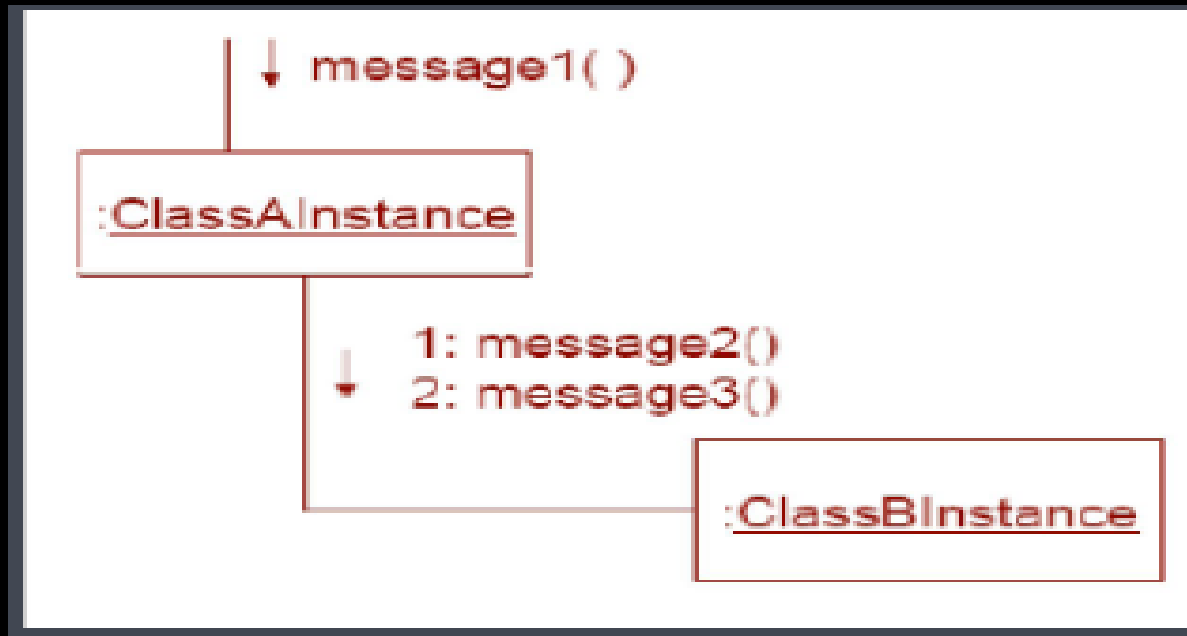
- Member of the Behavioral Group of diagrams
- Collaboration Diagrams captures dynamic behavior of the objects in the system.
- They are very useful for visualizing the relationship between objects collaborating to perform a particular task.

Collaboration Diagram

- Represents a Collaboration and Interaction
- **Collaboration** set of objects and their interactions in a specific context
- **Interaction** set of messages exchanged in a collaboration to produce a desired result
- Their purpose is to:
 - Model flow of control
 - Illustrate coordination of object structure and control

What it Represents?

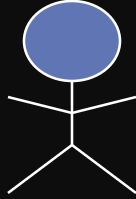



- Collaboration Diagrams illustrate object interactions in a graph or network format.



Collaboration Diagram Elements

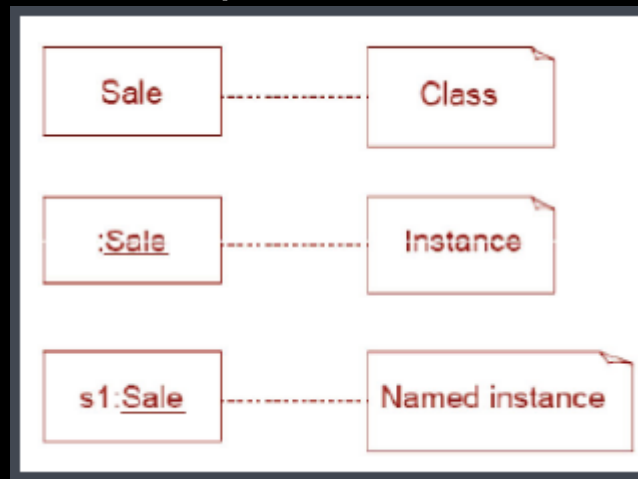
- There are three primary elements of a collaboration diagram:
 - Objects
 - Links
 - Messages

Collaboration Diagram Syntax

AN ACTOR	
AN OBJECT	
AN ASSOCIATION	
A MESSAGE	

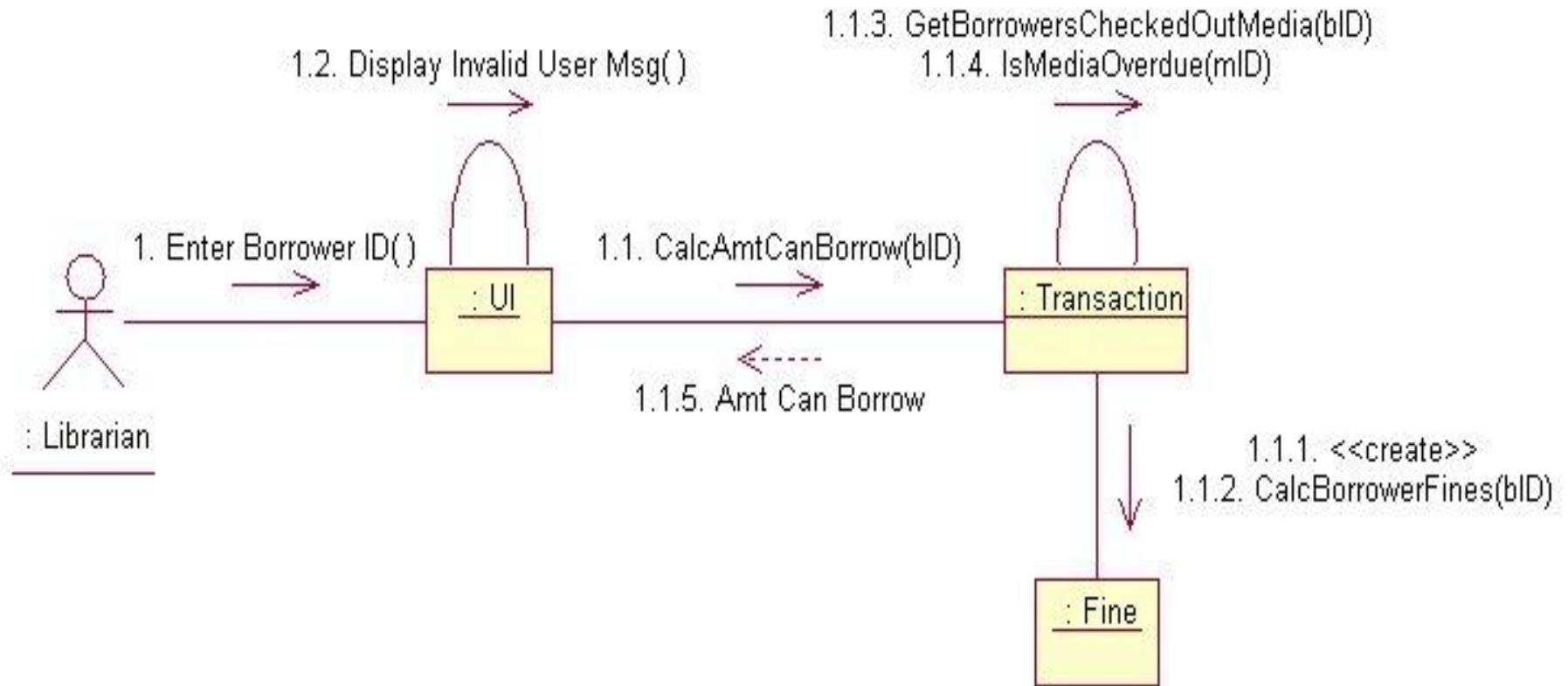
Notations used for Collaboration Diagrams-Objects

- **Objects** rectangles containing the object signature
 - **object name : object Class**
 - object name (optional) - starts with lowercase letter
 - class name (mandatory) - starts with uppercase letter
- Objects connected by lines ,actor can appear



Notations used for Collaboration Diagrams-Objects

- Objects participating in a collaboration come in two flavors—supplier and client
- **Supplier** objects are the objects that supply the method that is being called, and therefore **receive** the message
- **Client** objects call methods on supplier objects, and therefore **send** messages

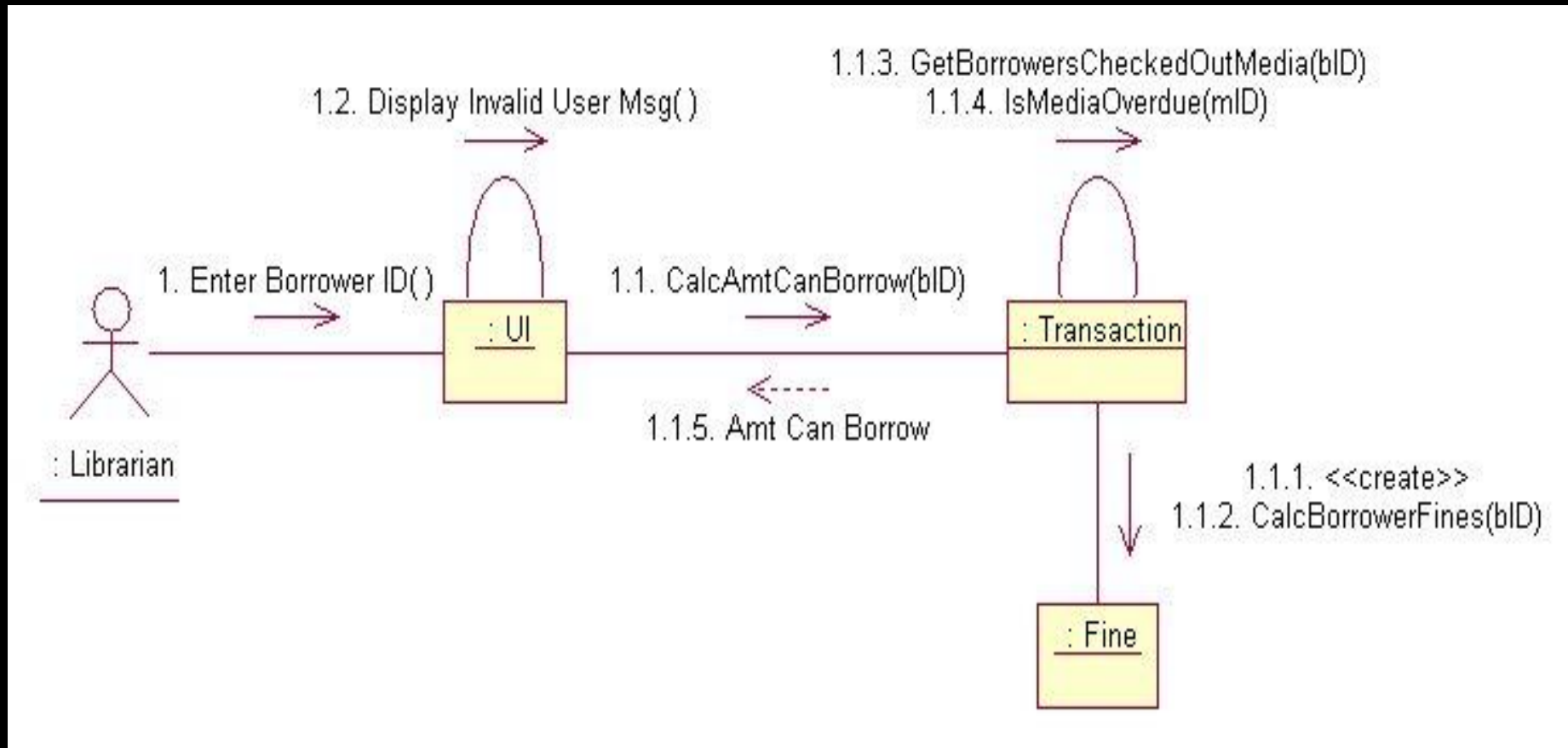


Transaction object acts as a Supplier to the UI (User Interface) Client object. In turn, the Fine object is a Supplier to the Transaction Client object.

Notations used for Collaboration Diagrams-Links

- The connecting lines drawn between objects are links
- They enable you to see the relationships between objects
- This symbolizes the ability of objects to send messages to each other
- A single link can support one or more messages sent between objects

Notations used for Collaboration Diagrams-Links



The visual representation of a link is a straight line between two objects. If an object sends messages to itself, the link carrying these messages is represented as a loop icon. This loop can be seen on both the UI object and the Transaction object.

Notations used for Collaboration

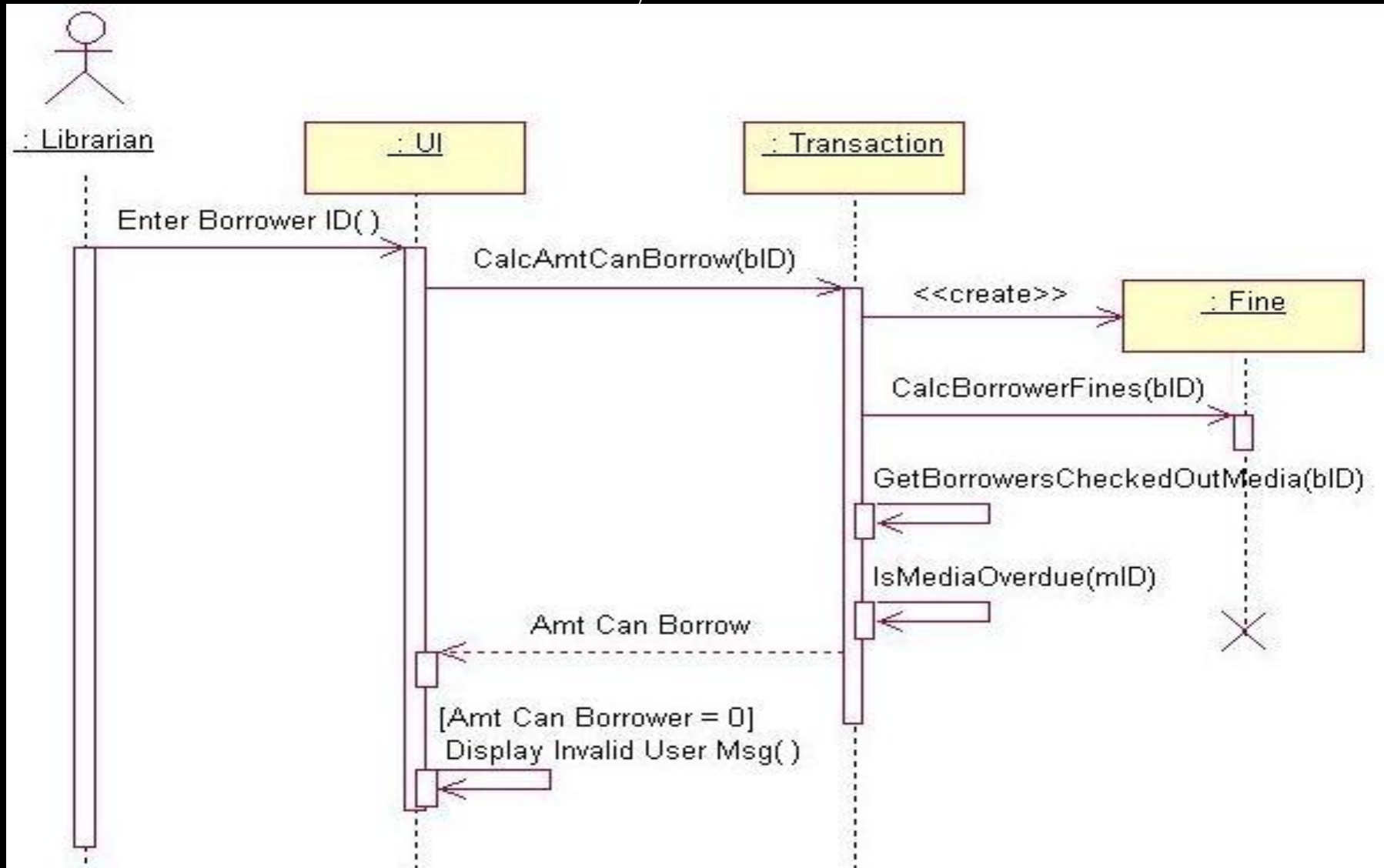
Diagrams- Messages

- An interaction between objects is implemented by exchanging messages.
- Messages in collaboration diagrams are shown as arrows pointing from the Client object to the Supplier object.
- Typically, messages represent a client invoking an operation on a supplier object
- Message icons have one or more messages associated with them.
- Messages are composed of message text prefixed by a **sequence number**
- Time is not represented explicitly in a collaboration diagram, and as a result the various messages are **numbered to indicate the sending order.**

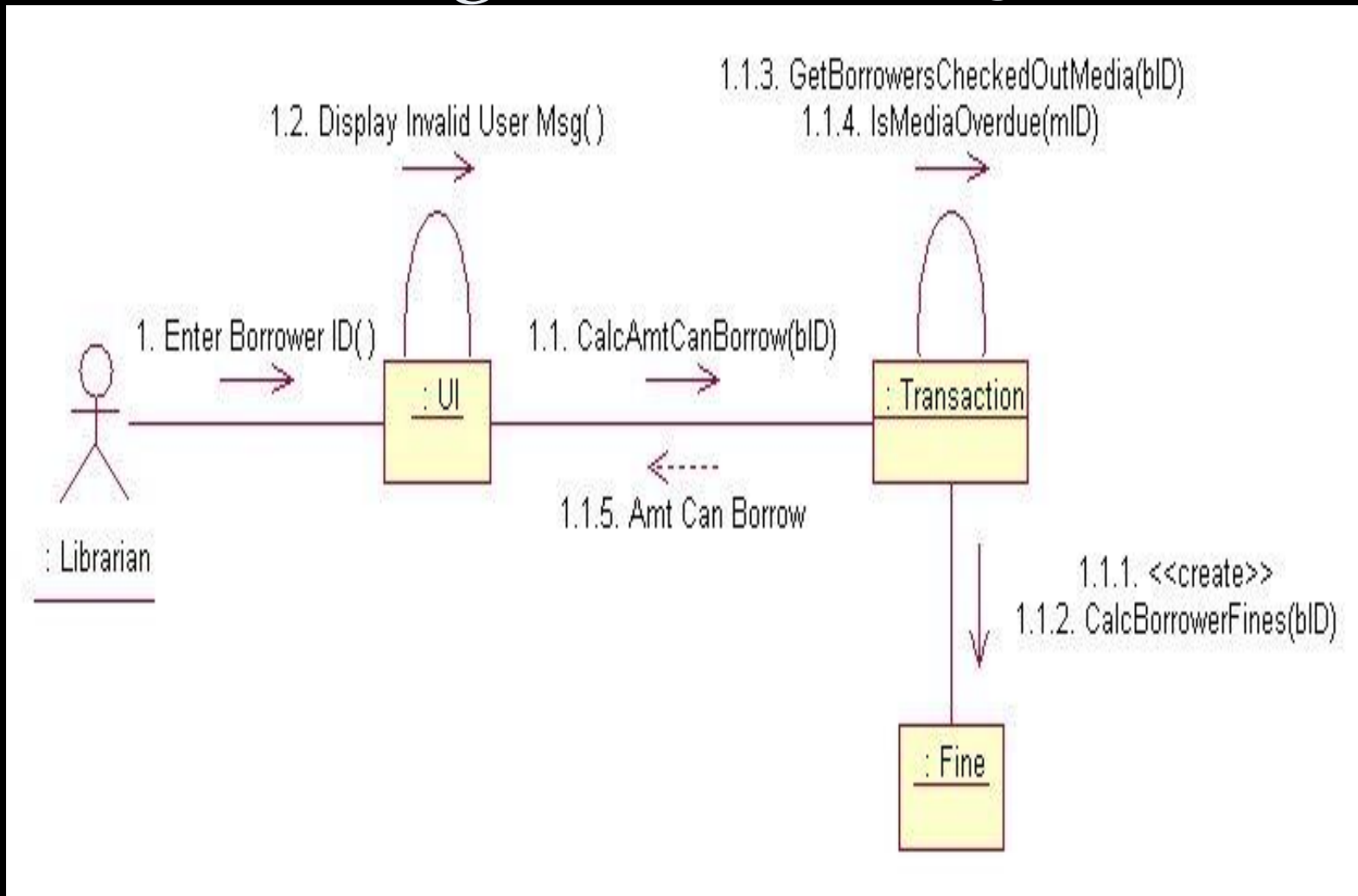
Example

- Here are the steps that occur in the use case named 'Borrow Book'.
- The librarian enters the borrower id in the system through UI to get the details about the number of books borrower can have.
- The UI calls the transaction object to calculate the amount that can be borrowed.
- Transactions objects creates and calculate the fine for borrower id .
- Borrowers checked out media and overdue media is also checked by transaction object and then returned to UI to displays this information for the librarian.
- If borrower's amount is equal to zero then UI displays the invalid message.

This particular sequence, shown in Figure , documents the interaction that occurs between business objects when determining how many items a borrower can check out of the library.



Notations used for Collaboration Diagrams-Messages



Iterating Messages

- Collaboration diagrams use syntax similar to sequence diagrams to indicate that either a message iterates (is run multiple times) or is run conditionally
 - An asterisk (*) indicates that a message runs more than once
 - Or the number of times a message is repeated can be shown by numbers (for example, 1..5)

Conditional Messages

- To indicate that a message is run conditionally, prefix the message sequence number with a conditional [guard] clause in brackets
 - [`x = true`]: [IsMediaOverdue]
- This indicates that the message is sent only if the condition is met.

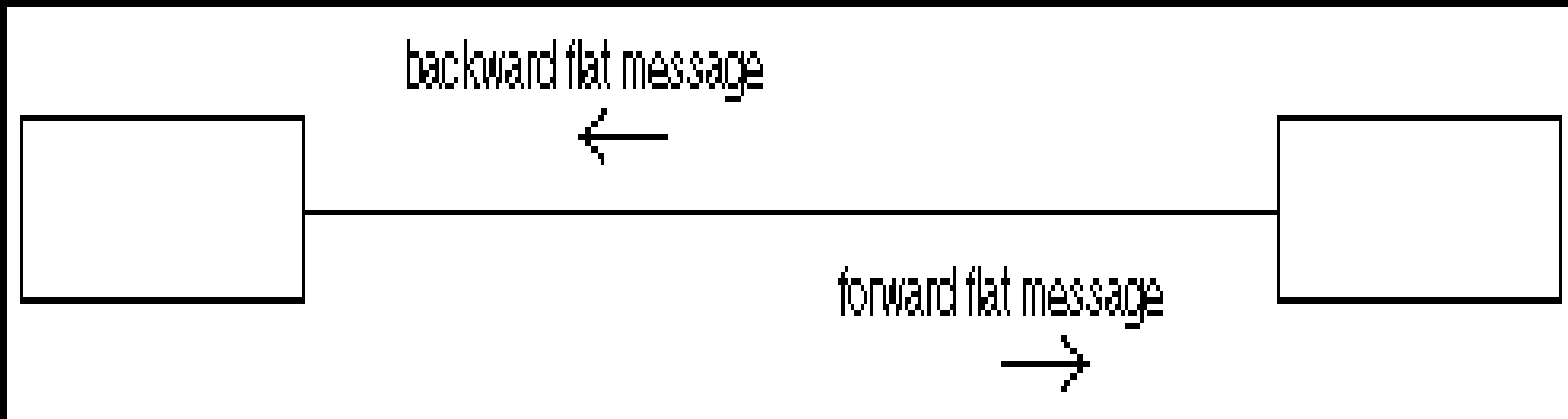
Nested Message

- The nested message represents a procedure call or other nested flow of control. The nested sequence is completed before the outer level sequence resumes. The nested message symbol is represented by a filled solid arrowhead.



Flat message

- The flat message shows the progression to the next step in a sequence. The flat message symbol is represented by a stick arrowhead.



Creation and Deletion

- Unlike sequence diagrams, you don't show an object's lifeline in a collaboration diagram.
- If you want to indicate the lifespan of an object in a collaboration diagram, you can use create and destroy messages to show when an object is instantiated and destroyed.

Objects Changing State

- State of an object can be indicated
- Initial state is indicated with <<create>>
- If an object changes significantly during an interaction, you can add a new instance of the object to the diagram, draw a link between them and add a message with the stereotype <<become>>

Change State of an Object



Steps to Creating a Collaboration Diagram

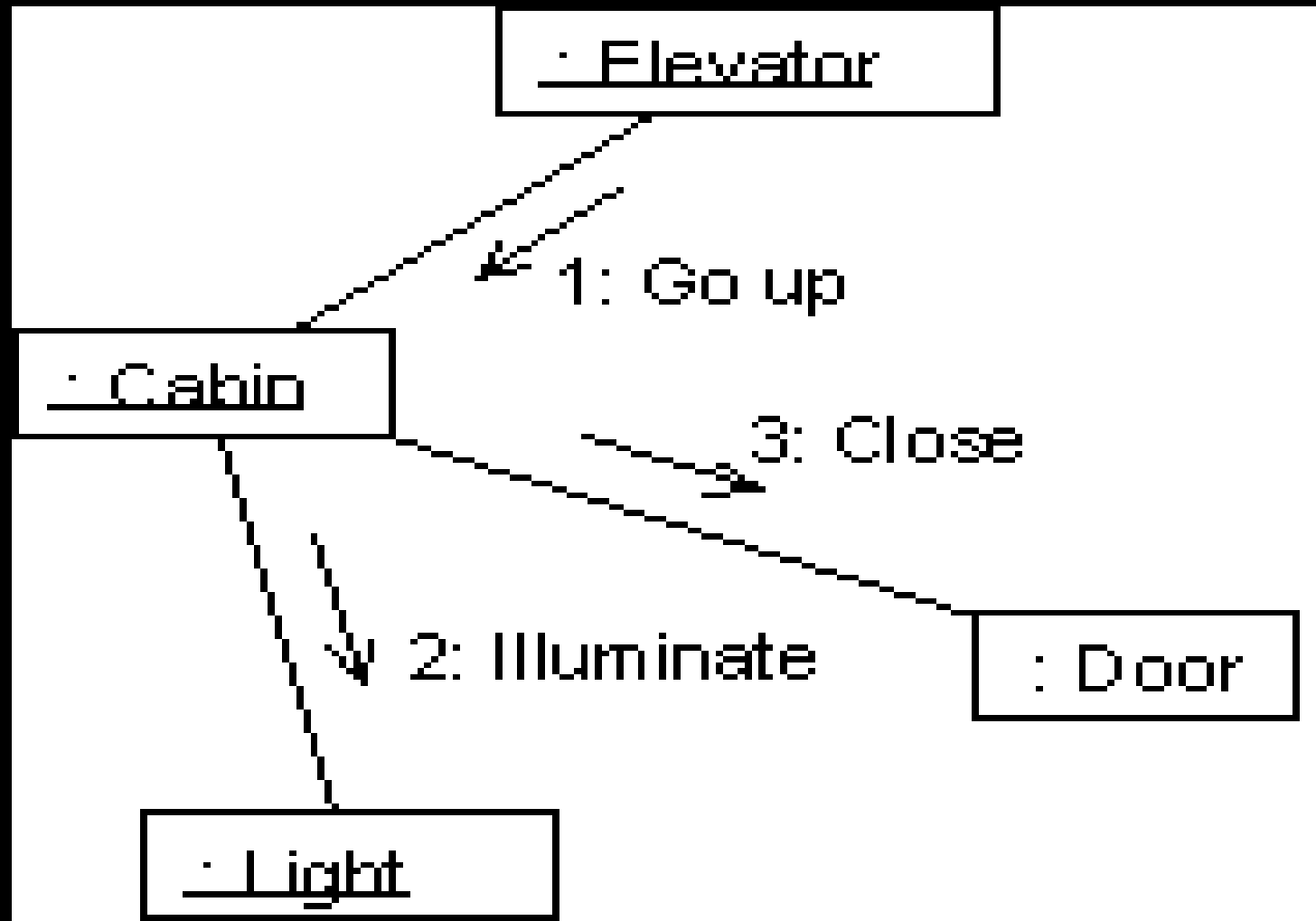
1. Determine the scope of the diagram- the use case it relates to.
2. Place the objects that participate in the collaboration on the diagram
 - o Remember to place the most important objects towards the center of the diagram.
3. If a particular object has a property or maintains a state that is important to the collaboration, set the initial value of the property or state.

Steps to Creating a Collaboration Diagram

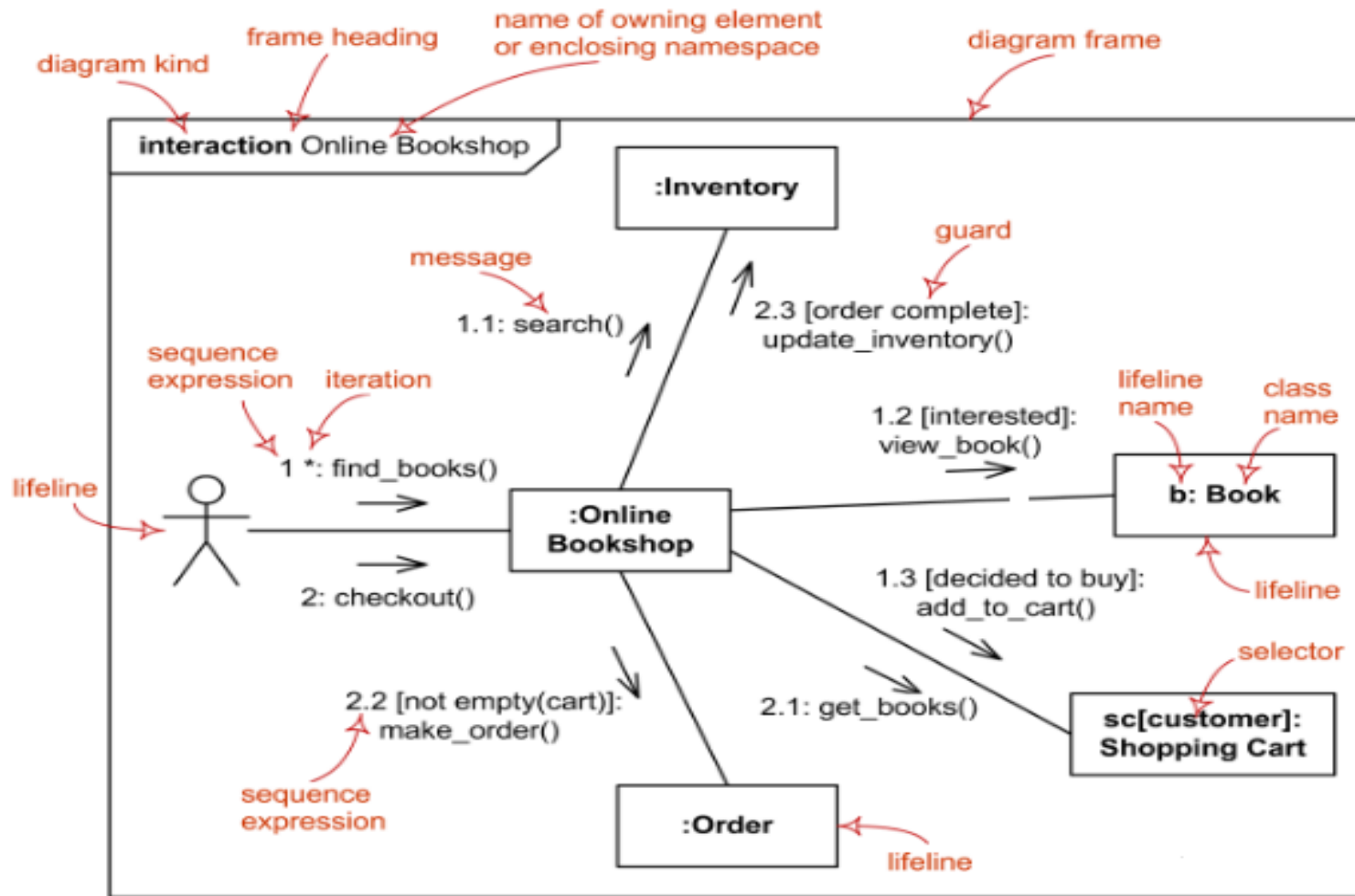
4. Create links between the objects
5. Create messages associated with each link
6. Add sequence numbers to each message corresponding to the time-ordering of messages in the collaboration

Collaboration Diagram Examples

Elevator- Collaboration Diagram



Online Book Purchase Collaboration Diagram



Collaboration vs Sequence Diagram

- In reality, sequence diagrams and collaboration diagrams show the same information, but just present it differently
- Not used as often as sequence diagrams but are closely related
- *Sequence Diagrams* are arranged according to Time.
- *Collaboration Diagrams* represent the structural organization of object.
- [Both sequence and collaboration diagrams are called *interaction* diagrams]

Collaboration vs Sequence Diagram

- In sequence diagrams, each message icon represents a single message.
- In collaboration diagrams, a message icon can represent one or more messages.
 - Notice between the Transaction and Fine objects - there is a single message icon, but there are two messages (1.1.1 and 1.1.2) associated with the icon.
- The difference between sequence diagrams and collaboration diagrams is that collaboration diagrams emphasize more the structure than the sequence of interactions.

Collaboration VS Sequence Diagram

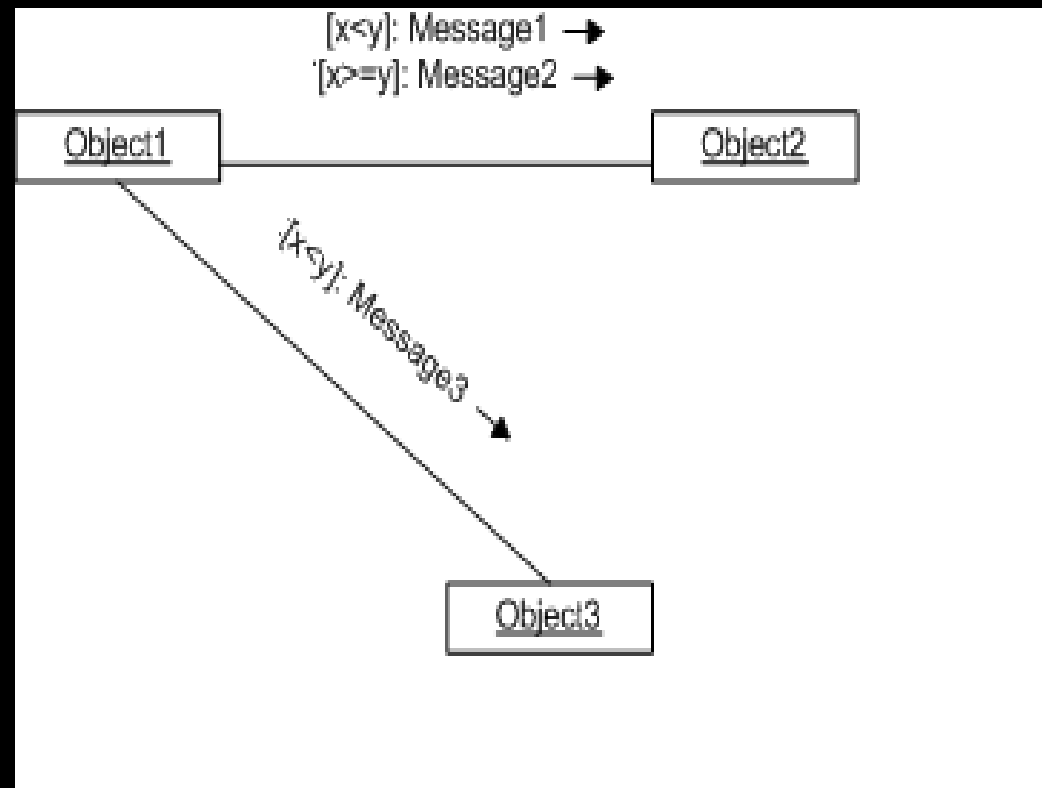
- Collaboration diagram illustrates object interactions in a graph or network format, in which objects can be placed anywhere on the diagram. It demonstrates how objects are statically connected.
- Sequence diagram illustrates interaction in a kind of fence format, in which each new object is added to the right. It generally shows the sequence of events that occur.

Why Collaboration Diagrams?

- Collaboration diagrams offer a better view of a scenario than a Sequence diagram when the modeler is trying to understand all of the effects on a given object and are therefore good for procedural design.
- A distinguishing feature of a Collaboration diagram is that it shows the objects and their association with other objects in the system apart from how they interact with each other. The association between objects is not represented in a Sequence diagram.

If-else

```
if (x<y)
{
object2.message1();
object3.message3();
}
else
{
object2.message2();
}
```

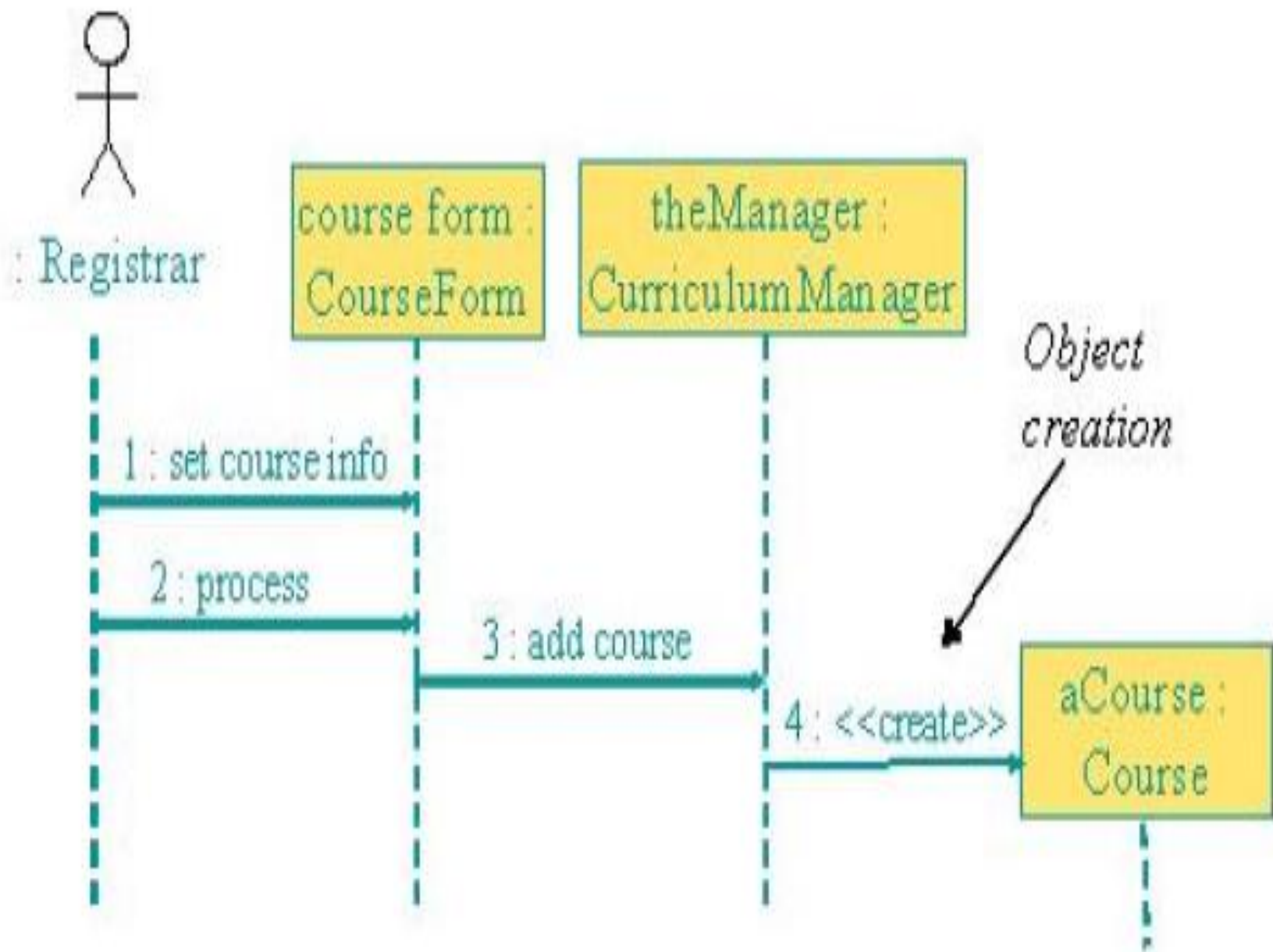


Collaboration diagrams – strengths and weaknesses

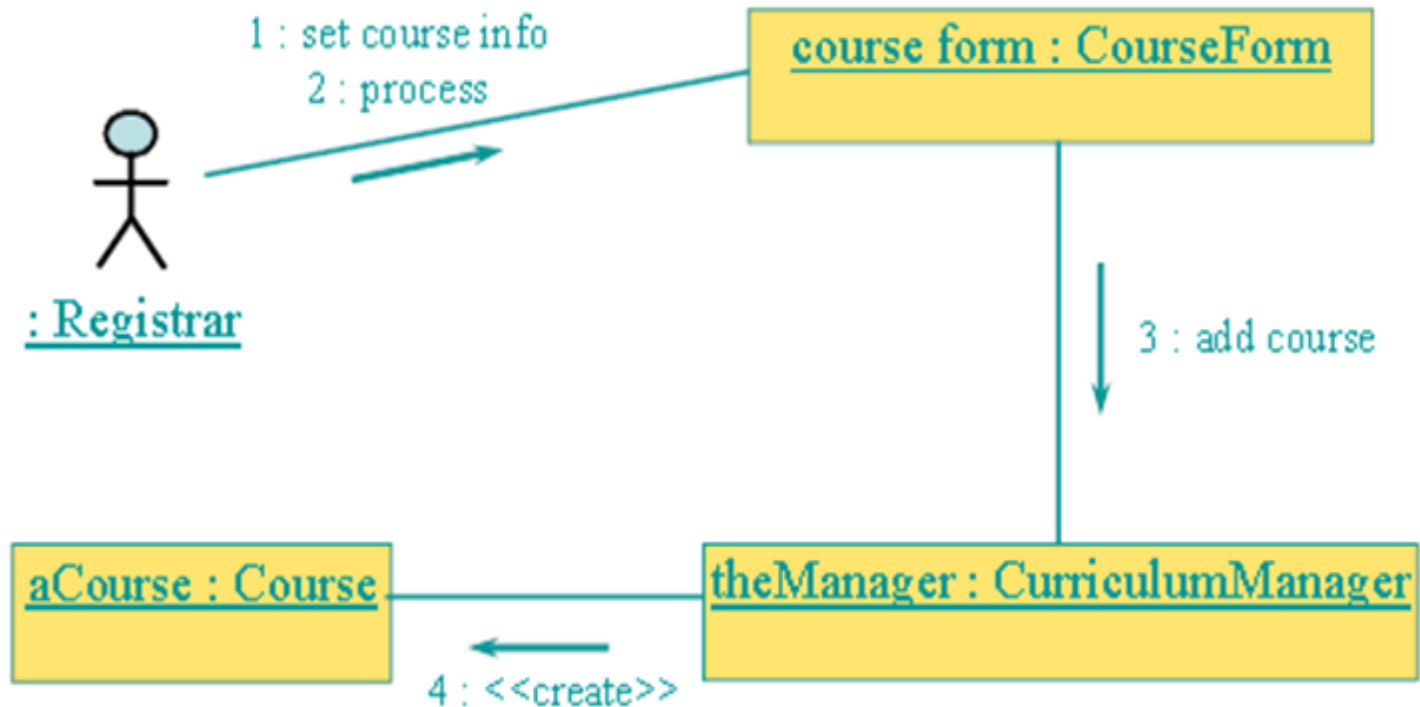
- One of the greatest strengths of collaboration diagrams is their simplicity.
- The principal weakness, however, is that although they are good at describing behavior, they do not define it. They typically do not show all the iteration and control that is needed to give an computationally complete description.
- **Strengths:**
 - they clearly show sequence or time ordering of messages.
 - the notation is rather simple
- **Weaknesses:**
 - forced to extend to right when adding new objects, consumes a lot of space

Class Activity

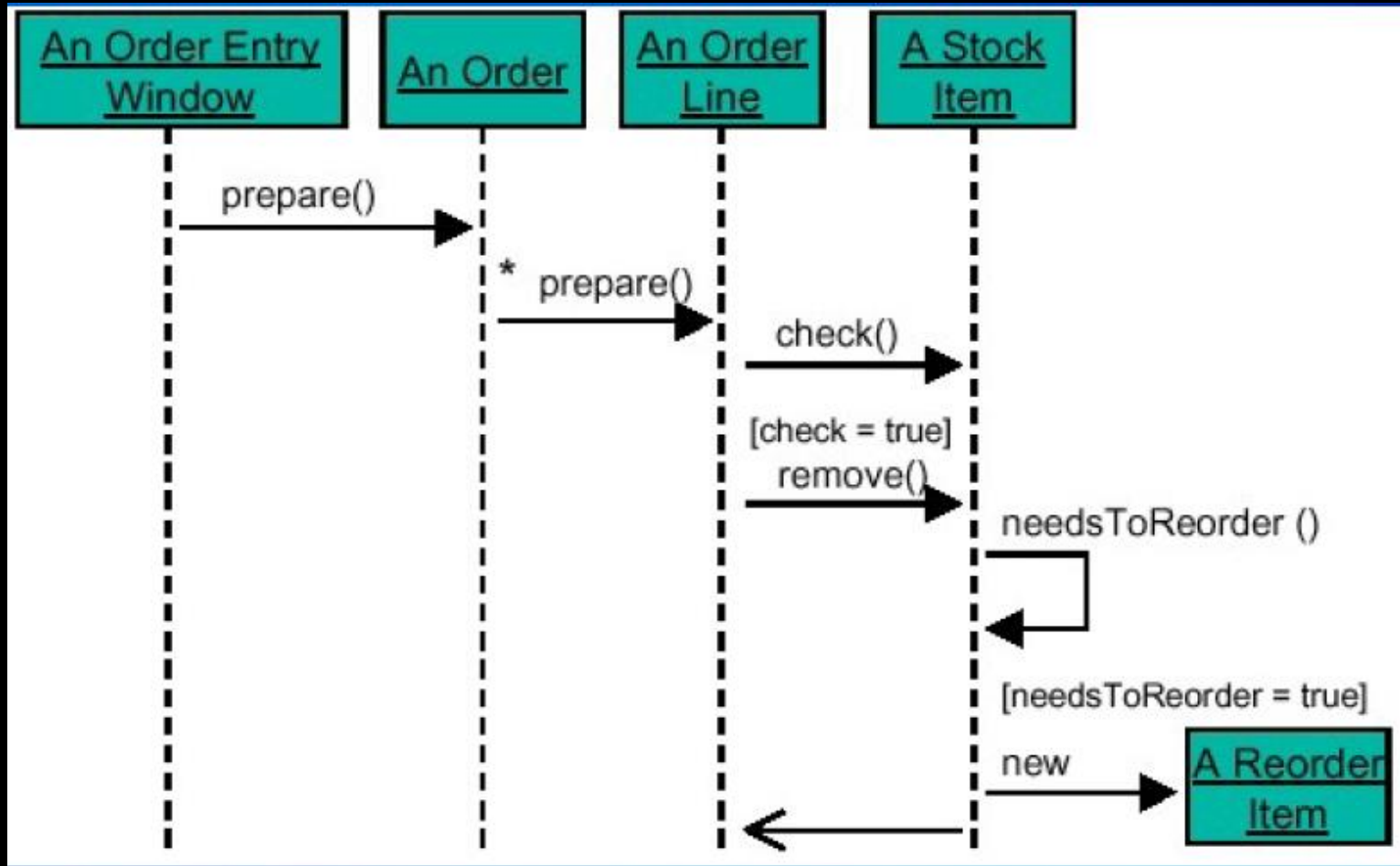
- Consider the software that controls a very simple cellular telephone. Such a phone has buttons for dialing digits, and a “send” button for initiating a call. It has “dialer” hardware and software that gathers the digits to be dialed and emits the appropriate tones. It has a cellular radio that deals with the connection to the cellular network. It has a speaker, and a display.
- **Use case: Make Phone Call**
 1. User presses the digit buttons to enter the phone number.
 2. For each digit, the display is updated to add the digit to the phone number.
 3. For each digit, the dialer generates the corresponding tone and emits it from the speaker.
 4. User presses “Send” for connecting to the network. The accumulated digits are sent to the network. The cellular radio establishes a connection to the network.
 5. The “in use” indicator is illuminated on the display
 6. The connection is made to the called party.



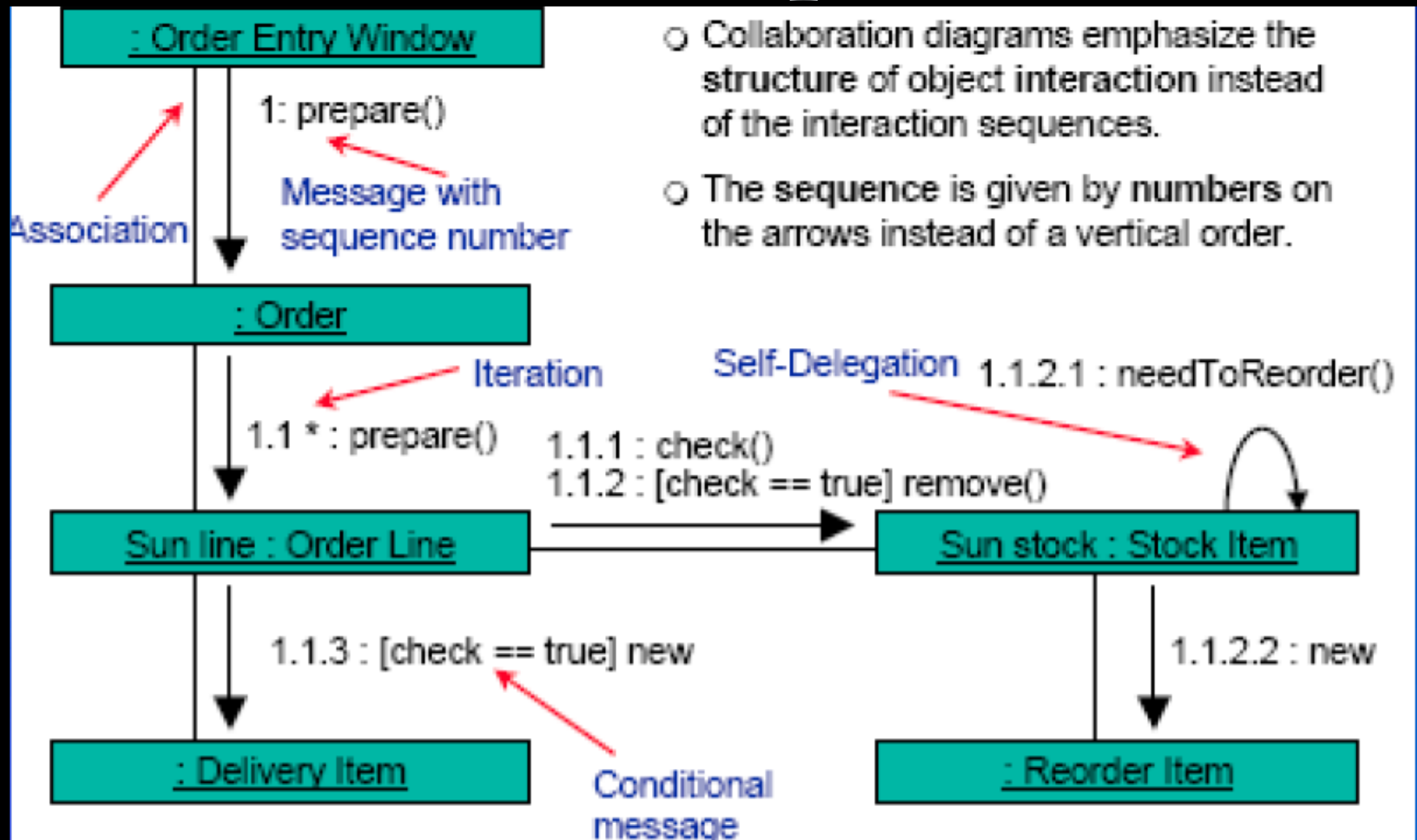
Example



Example



Example





That is all