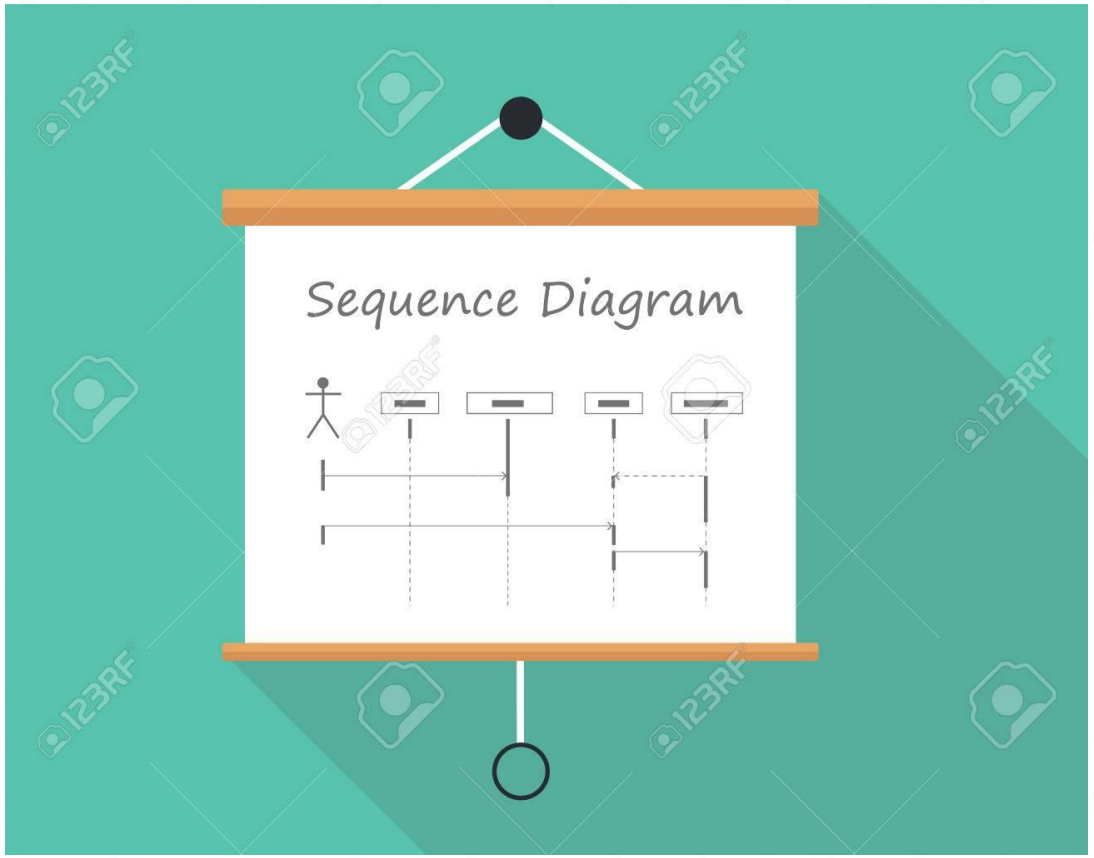
SLIIT

Discover Your Future

Software Engineering (IT2020)
2025

Lecture 2 - Interaction Diagrams – Part I

Interaction Diagrams

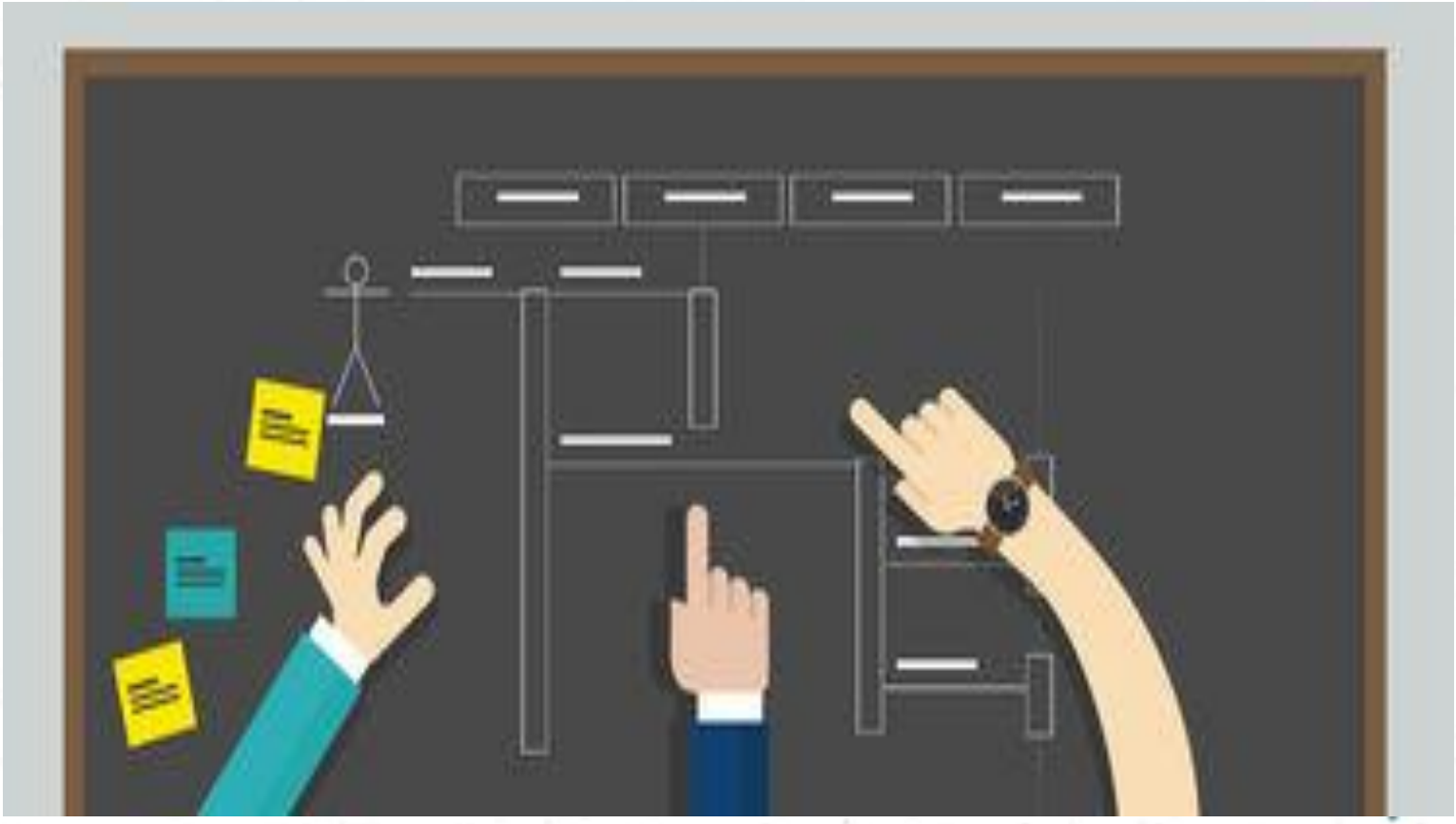


- An interaction diagram represents an interaction between the objects, which contains a set of objects and the relationship between them including the messages exchanged between the objects.

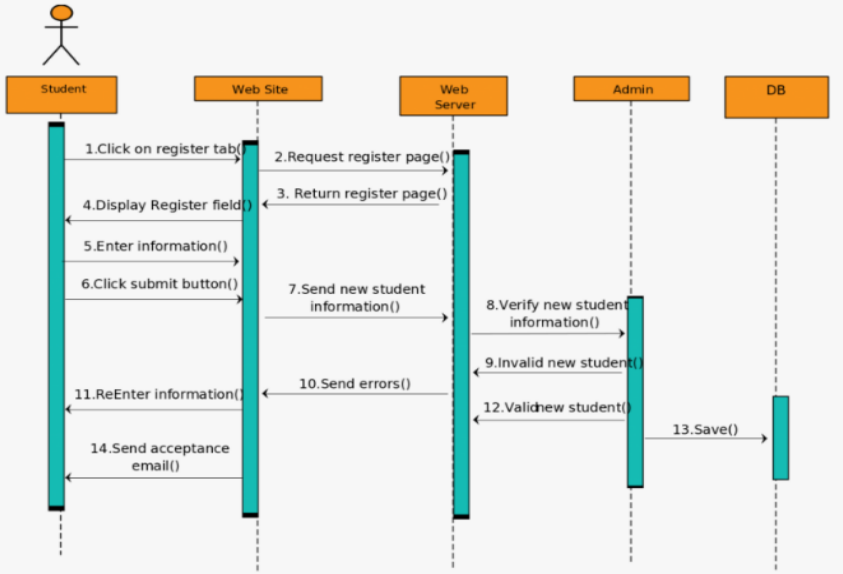
Interaction Diagrams

- Describes how groups of ***objects*** interact in some behavior.
- Typically, captures the behavior of a single use case.
- Two kinds of interaction diagrams:
  - Sequence diagrams
  - Communication diagrams-(Collaboration diagram)

Sequence Diagram



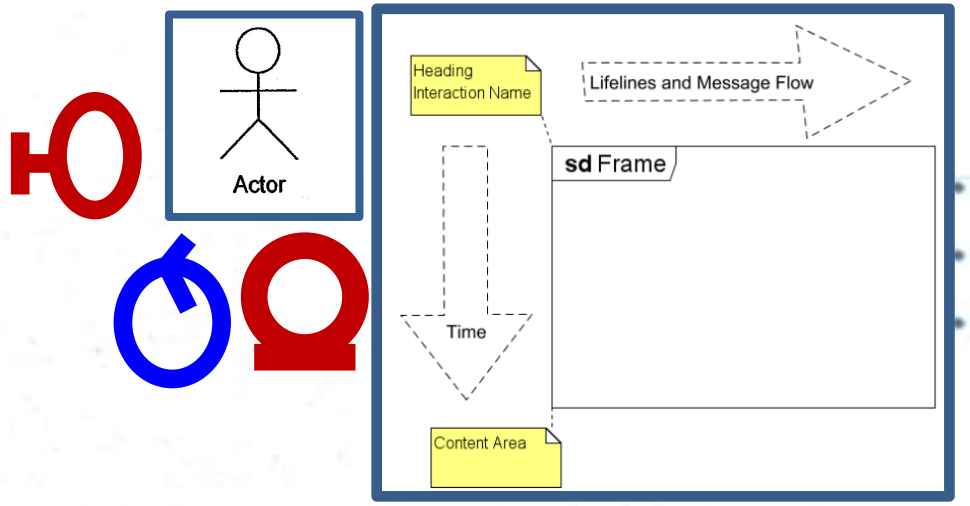
What is a Sequence Diagram?



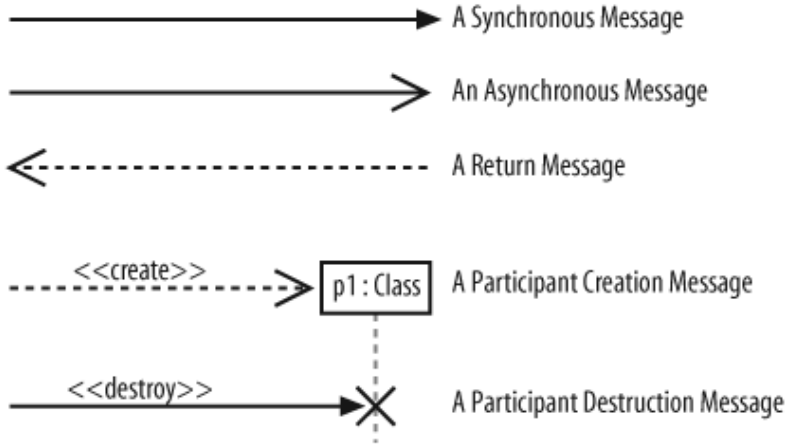
- A sequence diagram is one of the two interaction diagrams.
- The sequence diagram emphasizes on the time ordering of messages.
- In a sequence diagram, the objects that participate in the interaction are arranged at the top along the x-axis.

Elements of a Sequence Diagram

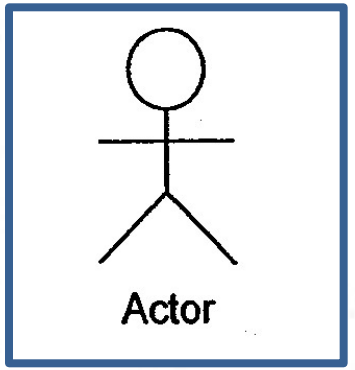
- Frame
- Actor
- Objects
  - Stereotypes
- Lifeline
- Execution bar (Activation bar)
- Messages
- Complex Interactions (Tags)



Object name : Class name



Actor



- Actors are the participants in a sequence through sending and/or receiving messages.
- External to the subject.
- Represent roles played by human users, external hardware, or other subjects.

Objects

- Objects are shown in the same way as in UML object diagrams.
- A named object

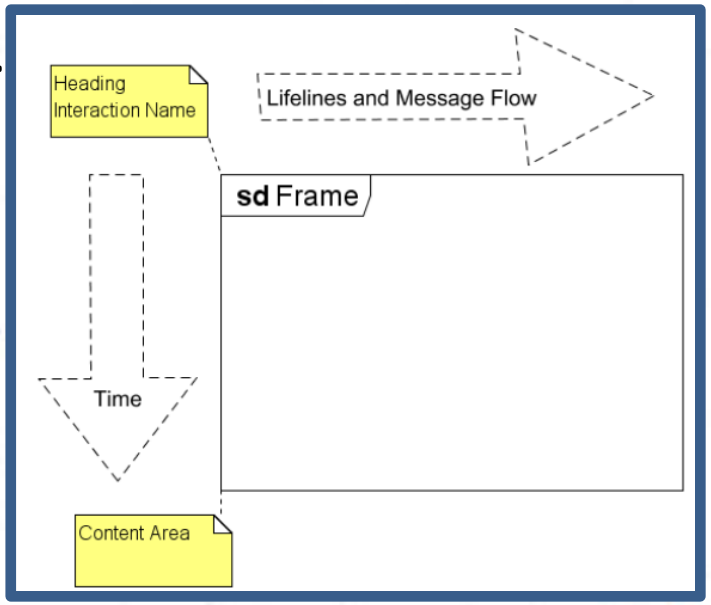
  (Named instance of a class)

- An anonymous object

  (Anonymous instance of a class)

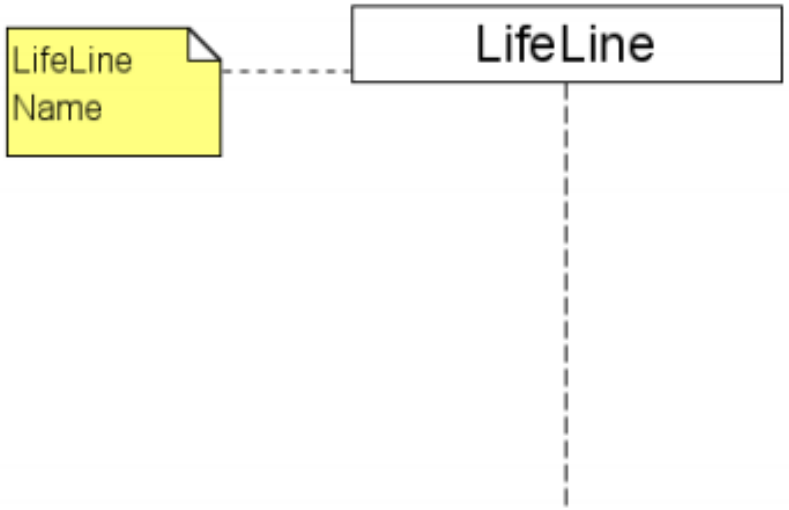
Frame

- A frame represents an interaction, which is a unit of behavior that focuses on the observable exchange of information between different objects.
- **sd** → Sequence Diagram <sd name of the diagram>



Lifelines

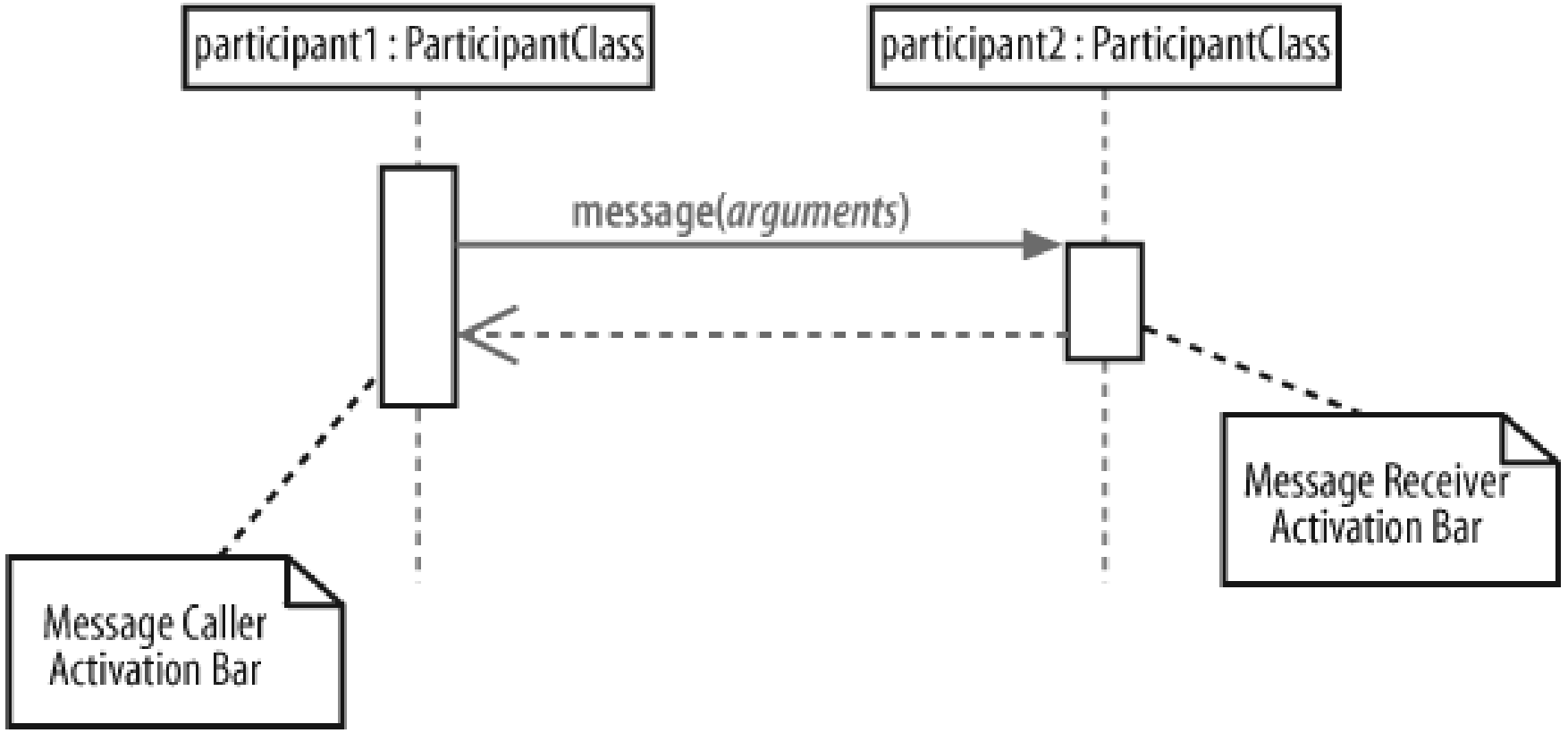
- Sequence diagrams are organized according to the time.
- Each participant has a corresponding lifeline.
- Each vertical dotted line is a lifeline, representing the time that an object exists.



Execution/Activation Bar

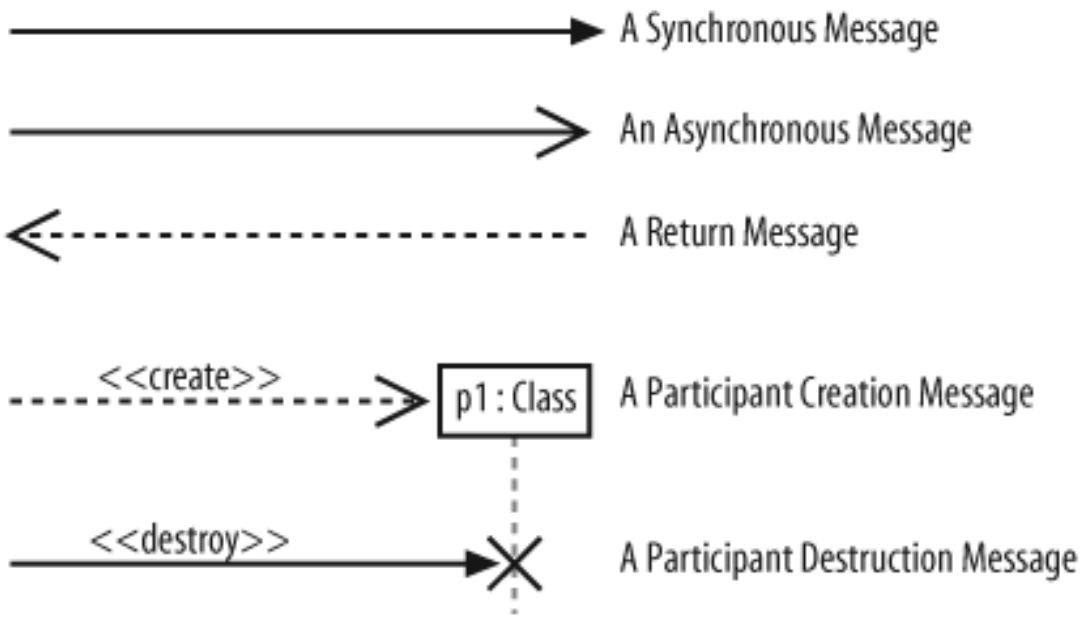
- When a message is passed to a participant it triggers or invokes, the receiving participant into doing something; at this point, the receiving participant is said to be active.
- To show that a participant is active, i.e., doing something, you can use an activation bar, as shown next.

Execution/Activation Bar cont...



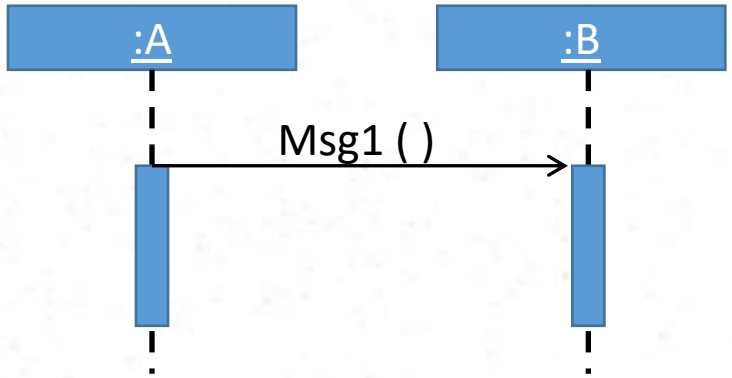
Messages

- Messages are used to illustrate communication between different active objects of a sequence diagram.



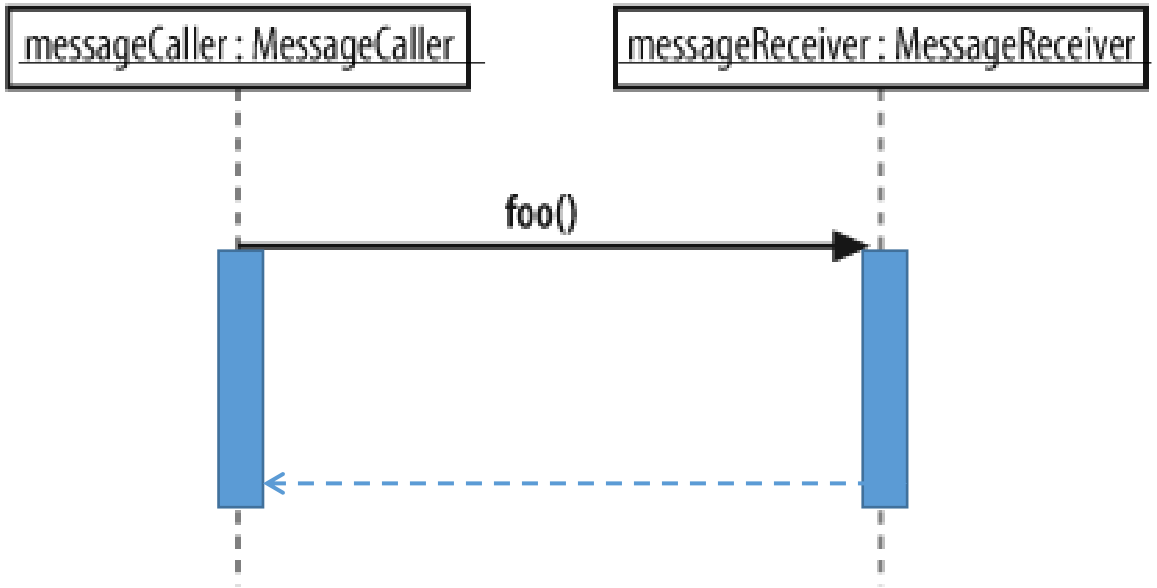
Message Passing

- Message passing is the invocation of a method in one object, by another method that belongs to a different object.
- When a message is sent from object A to object B:
  - object A asks object B to execute one of its methods (i.e. invoke the method);
  - object B receives the message and executes the method;



Synchronous Call

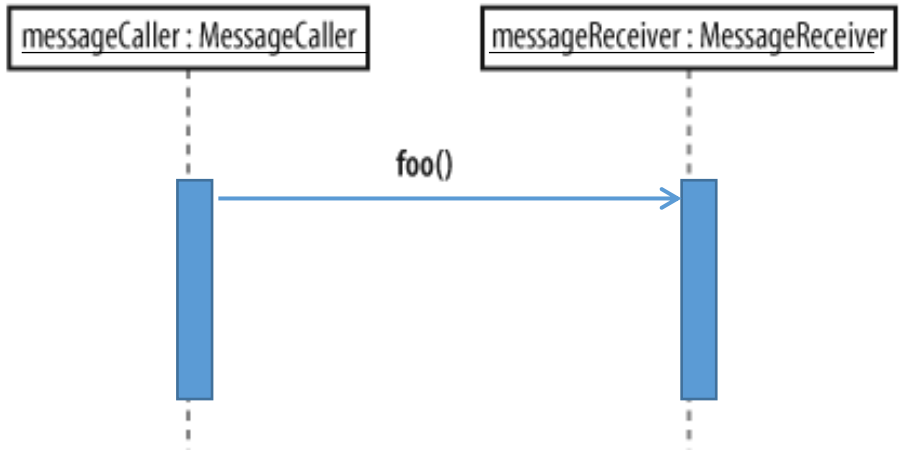
- Synchronous message is used when the sender waits until the receiver has finished processing the message.



Asynchronous Call

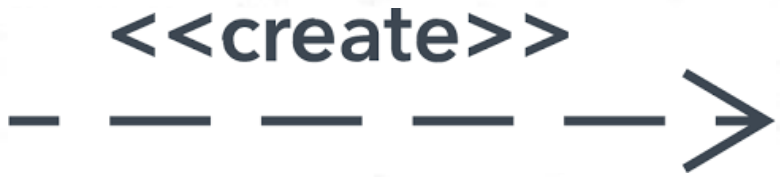
- **Asynchronous call** - The sender does not wait for the receiver to finish processing the message (for any return values), it continues immediately.

- An open arrowhead is used to asynchronous call.



Create and Delete Messages

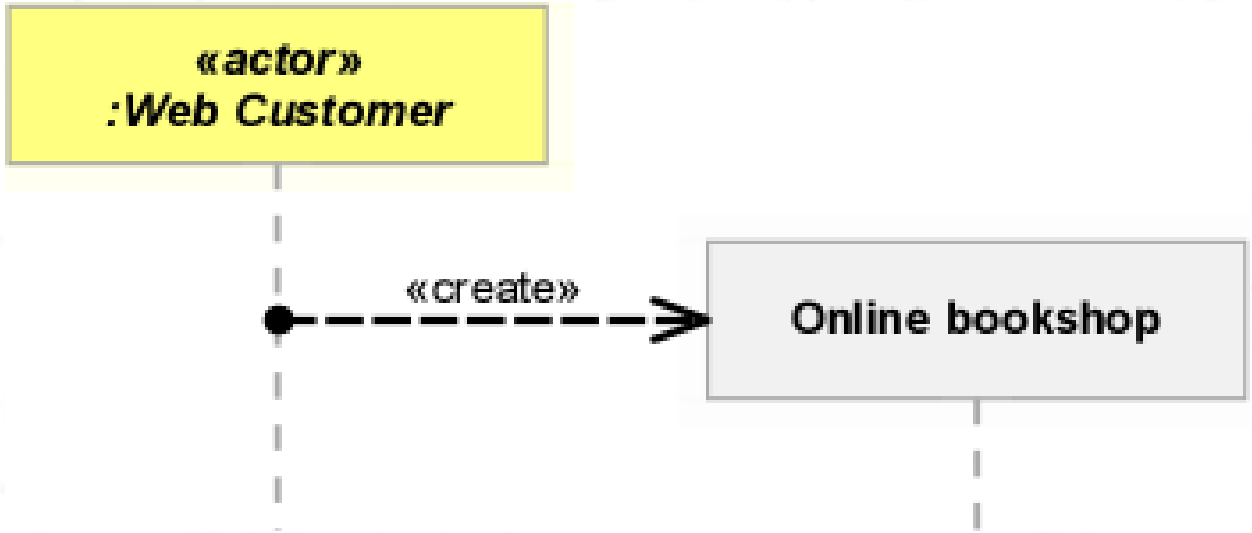
- **Create Message**
  - Create message to instantiate a new object in the sequence diagram.
  - There are situations when a particular message call requires the creation of an object.
  - It is represented with a dotted arrow and create word labelled on it to specify that it is the create Message symbol.

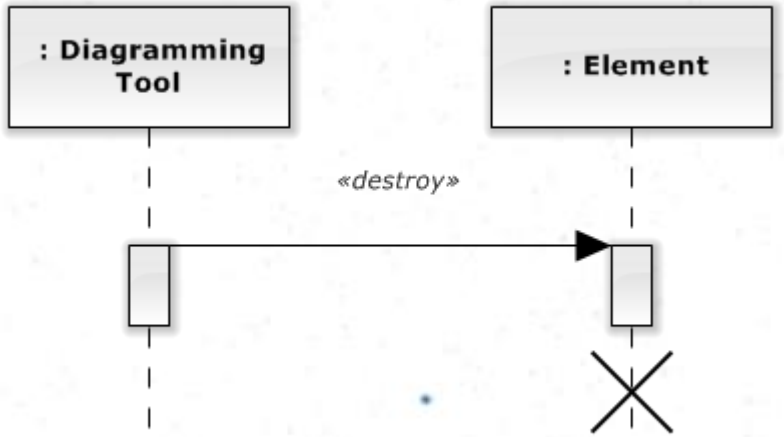


Create and Delete Messages cont…

- **Delete Message**
  - Use to remove / delete unwanted objects.
  - In a garbage-collected environment, you don't delete objects directly, but it's still worth indicating when an object is no longer needed and is ready to be collected.

Create and Delete Messages cont…





Lost and Found messages

- **Lost Message :**
  - A lost message occurs when the sender of the message is known but there is no reception of the message.
  - This also allows the modeler to consider the possible impact of a message's being lost. (This message allows advanced dynamic models to built up by fragments without complete knowledge of all the messages in the system.)

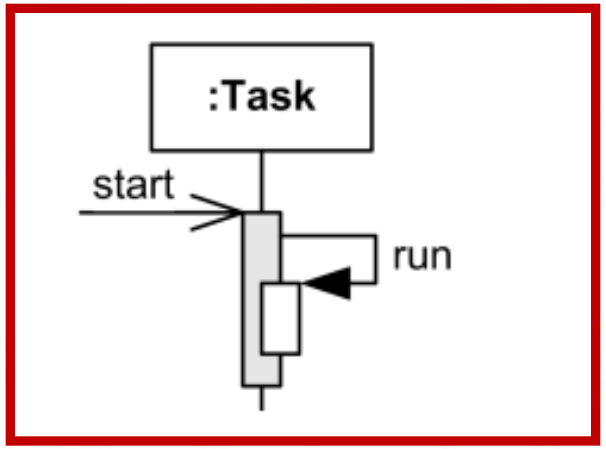
Lost and Found messages

- **Found Message:**

  Found message indicates that although the receiver of the message is known in the current interaction fragment, the sender of the message is unknown.

Self Calls

- A **self message is** one method calling another method belonging to the same object. (message to the same lifeline)
- It is shown as creating a nested focus of control in the lifeline's execution occurrence.
- Represented by overlapping rectangles on the same lifeline.



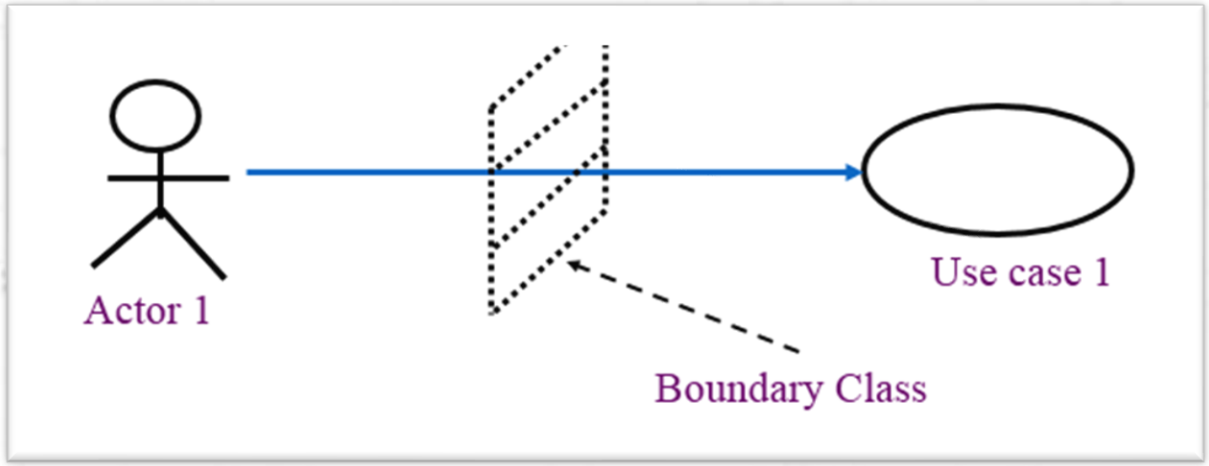
Class / Object Stereotypes

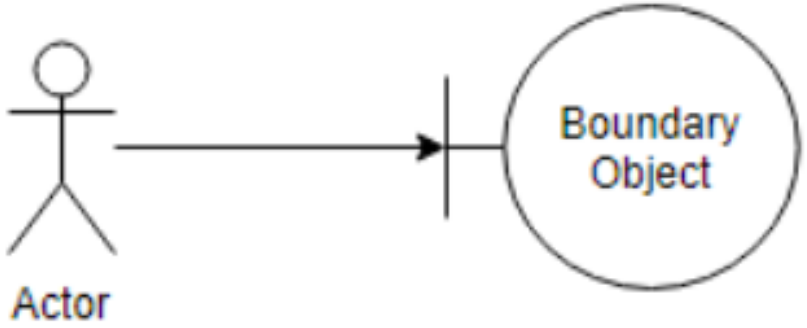
- Stereotypes differentiate the roles that classes / objects can play.
- Classes or objects can be classified into one of the following three stereotypes:
  - **Boundary** classes / objects - model interaction between the system and actors (and other systems)
  - **Entity** classes / objects - represent information and behaviour in the application domain
  - **Control** classes / objects - Co-ordinate and control other objects

Boundary Classes / Objects

- Boundary is a stereotyped class or object that represents some system boundary.
  - e.g. a user interface screen, system interface .
- The interaction often involves inputs and outputs.
- It is often used in sequence diagrams which demonstrate user interactions with the system.
- This is a class / object used to model the interaction between the system and its actors.
- To find the Boundary classes, you can examine your use case diagram and scenario.
- At a minimum, **there must be at least one *Boundary* class / object for every actor-use case interaction**.

Boundary Classes / Objects cont…



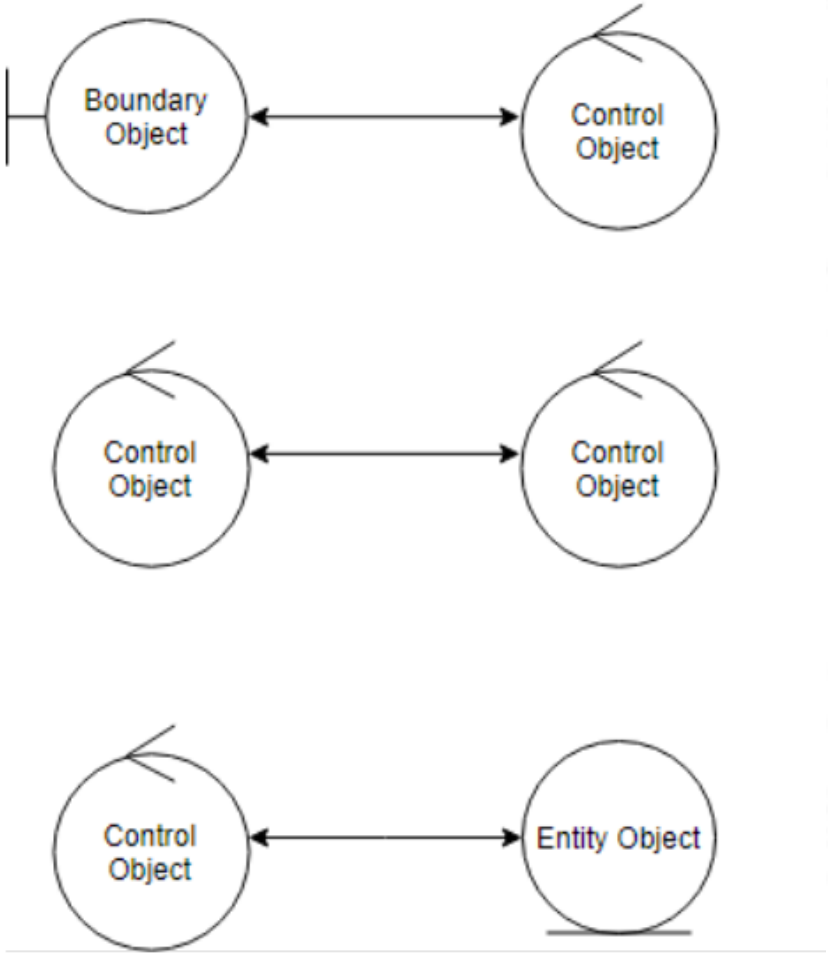


Control Classes / Objects

- Control classes represent coordination, sequencing, transactions and control of other objects
- It co-ordinates the events needed to realize the behaviour specified in the use case.
- Also, models complex computations and algorithms.
- Typically, they coordinate the movement (parameters) between boundary and entity classes.
- **There is at least one control class per use case**
  - Eg. Running or executing the use case.

- One or several control classes could describe use case realization.

Control Classes / Objects cont…



Entity Classes / Objects

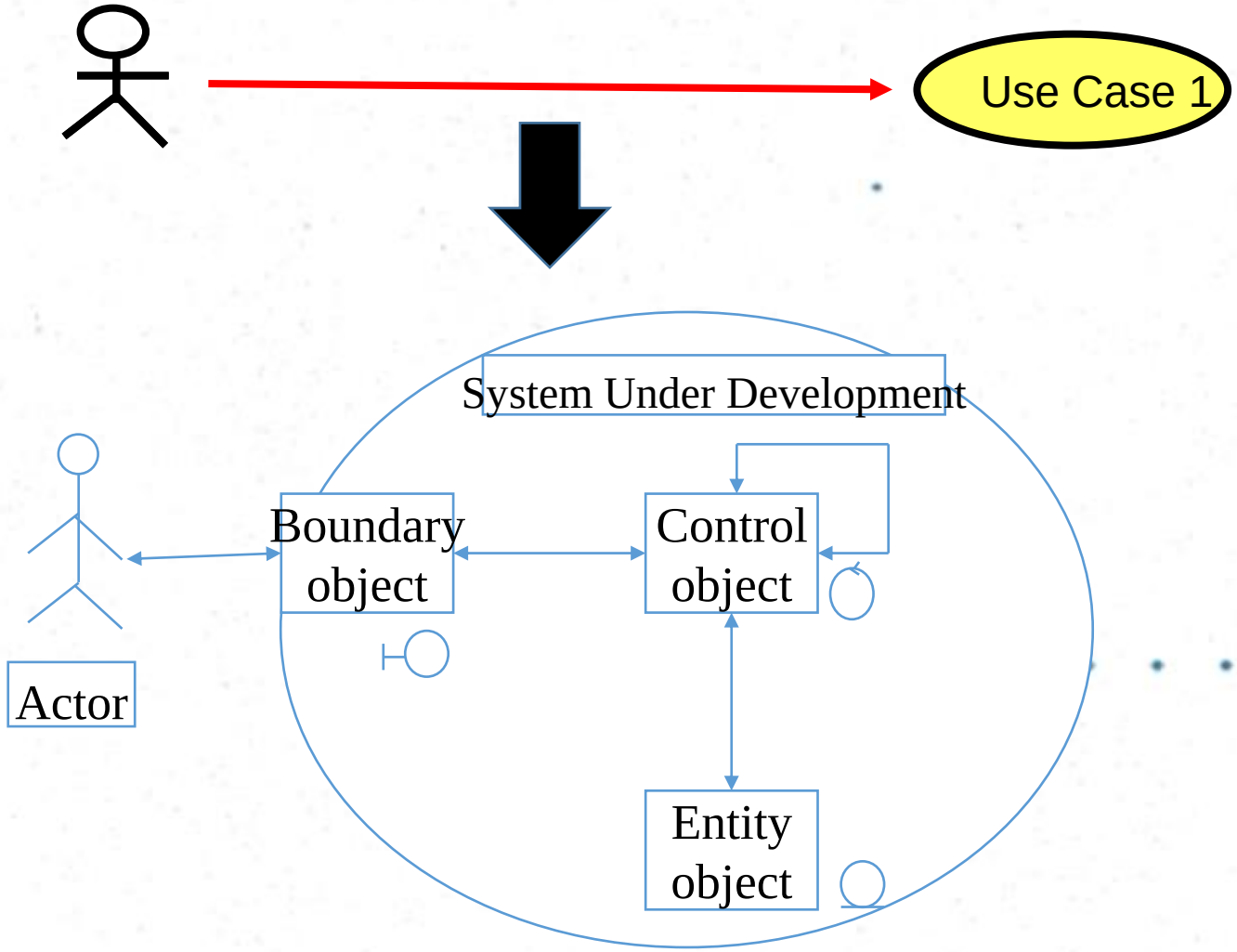
- Entity is a stereotyped class or object that represents some information or data.
- These classes are needed to perform tasks internal to the system.
- It reflects a real-world entity.

  Eg:

  - Account
  - Student
  - Lecturer, etc...

Entity Class / Objects cont…

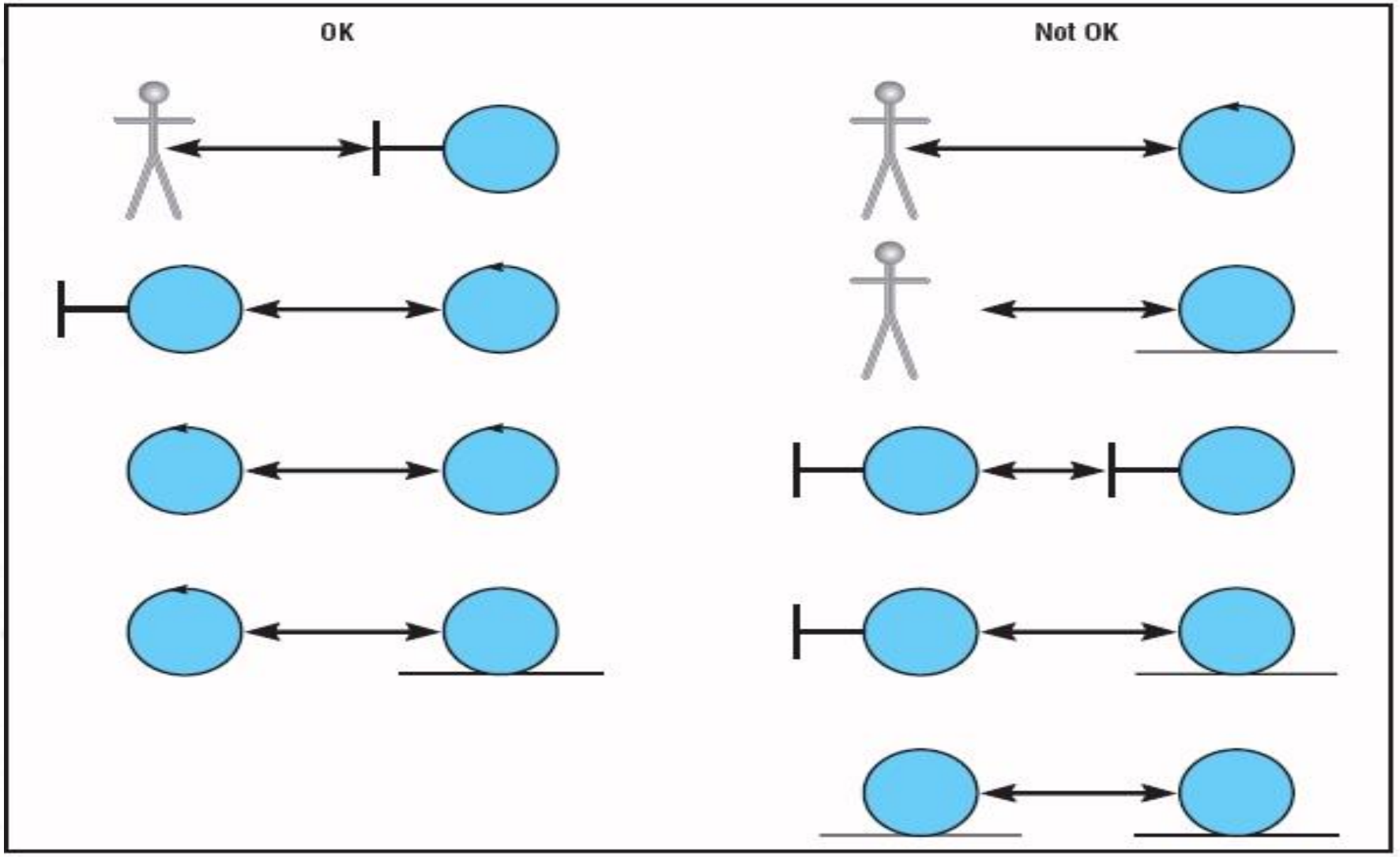
Block Diagram: Class / Object Stereotypes



How to Identify Class / Object Stereotypes

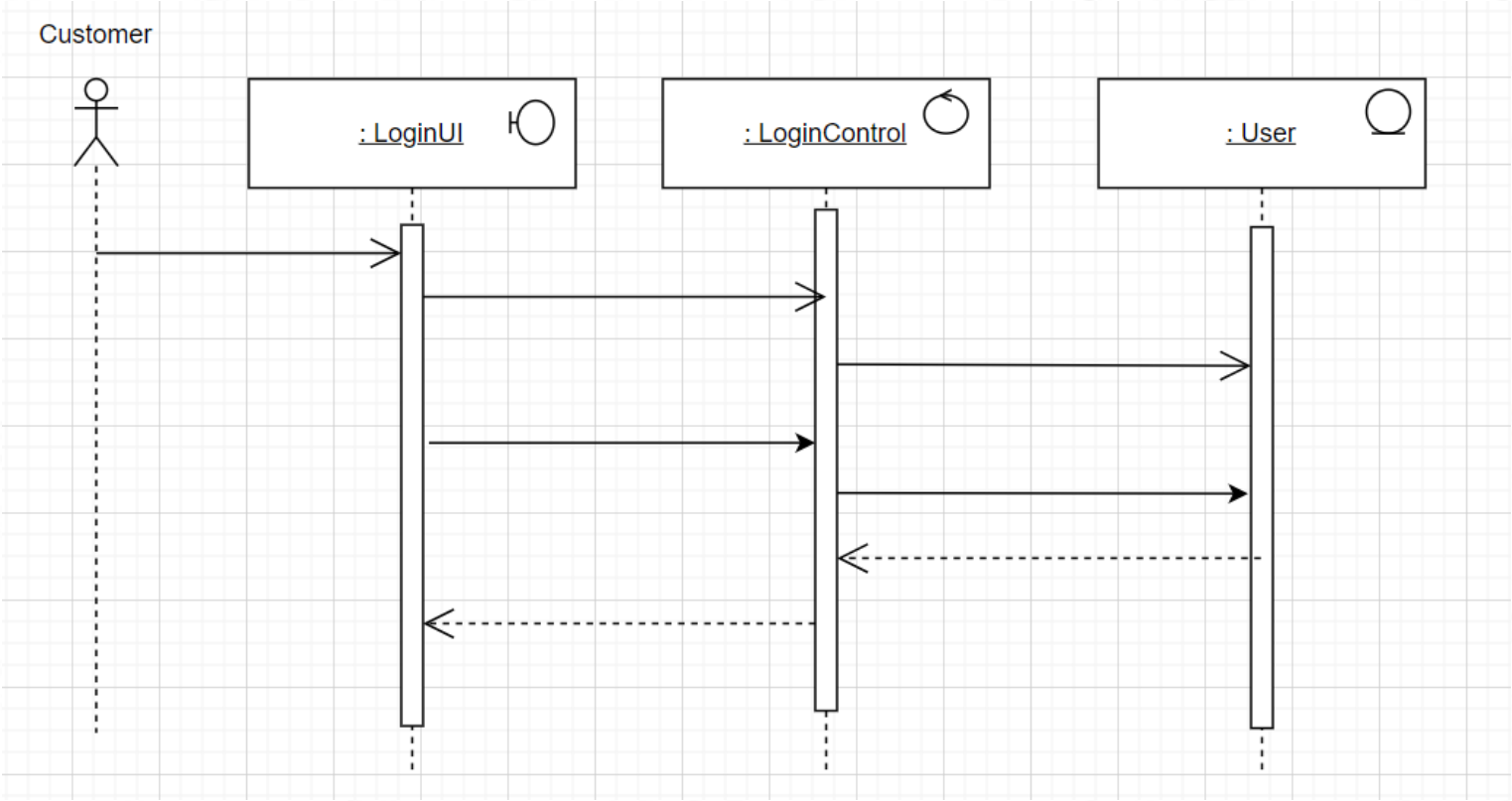
- Examine the models of a Use Case and identify **Entity classes** by examining the **data(information) storage requirements** of the Use Case.
- **Boundary classes** can identify by examining the **point of interaction of the actor with the system**.
  - One central **boundary class** for each actor. (primary window).
- **Control classes** can be identified by examining the **business process of the use case**.
  - One or more control class/es responsible for handling the control of the Use Case.

Rules of Stereotype Usage



Stereotype Representation in Sequence Diagrams

- Each object in the sequence diagram better to indicate its stereotype and the objects should communicate as specified in the stereotype rules.



Activity 1

Draw a sequence diagram for a login process of an online ordering system.

Note: You may use suitable boundary, control and entity classes.

- Customer login using his/her username and password.
- System request user details from the user DB and validates the username and password by comparing entered data with the user details in the user DB.
- Then the system will display status of the user validation to the customer.

Activity 2

The sequence of steps carried out in the "**Manage course information**" flow are given below. Draw a sequence diagram for the given description.

- The system administrator wants to do some changes to a course. He clicks on the Manage Course icon in Manage Course UI which in turn invokes the ManageCourse functionality of CourseControl class.
- The ManageCourse function of the CourseControl class first modifies the topic of the course through TopicModification function of the Topic class.
- Then, ManageCourse functionality invokes CourseModification functionality of a Course.
- Finally, the system administrator invokes AssignTutor function of the Tutor, to assign a tutor to the selected course.

Note:

- Assume all are Asynchronous messages.
- Use suitable boundary, control and entity classes.

Activity 3

Draw a sequence diagram for the given "Inquire book status" scenario by identifying stereotypes.

Name: Inquire book status

Brief Description: Handle from the borrower about availability of a book.

Actor: Librarian

Flow of Events:

1. Library member clicks "SerachBook" option in the Library UI by giving the Book ID.
2. Then LibraryController class will call SearchBook method in the Book class.
3. Then the system will send book status to the library member.
4. Next, the library member click reserve option in the UI and the LibraryController will call ReserveBook function of Book class.
5. Finally, Book class will send the reserve status to the library member.

References

- UML 2 Bible
  - Chapters 8 & 9
- Learning UML 2.0 by Kim Hamilton, Russ Miles
- Applying UML and Patterns by Craig Larman
- Chapter 15UML 2 Bible
  - Chapters 8 & 9
- TheElementsofUML2Style
  - Chapter 7

Thank you!