Sequence Diagrams

2021-’22 Winter SWE

Design - Text Book Approach

Object Oriented Design [OOD] predominantly bottom up approach.

Object Orientation

Identify the classes

- Their instances are Objects
- They interact to capture the functionalities

Simple to Complex

The Bottom Up Approach

Object Orientation - Learn About

Abstraction - Procedural and Data

Encapsulation

Inheritance - is-a / has-a relationships

Polymorphism - overloading / overriding

Importance of INTERFACEs

Object Orientation

Many terms which can be confusing

Java or C++

One example for each with practical code snippet

Unified Modeling Language

Agree on OOD

Different ways of going about - Different notational systems

Booch - Rumbaugh - Jacobson

Object Management Group (OMG) adopted UML in 1997, ISO in 2005

A standard representation of the Object Oriented Design [meaningful skills]

Unified Modeling Language

A standard representation of the Object Oriented Design including

- Activities
- Components
- Interfaces
- Interactions

Tools to draw these, and these days moving towards

Automatic Code Generation

UML Diagrams

Structure Diagrams

Behavior Diagrams

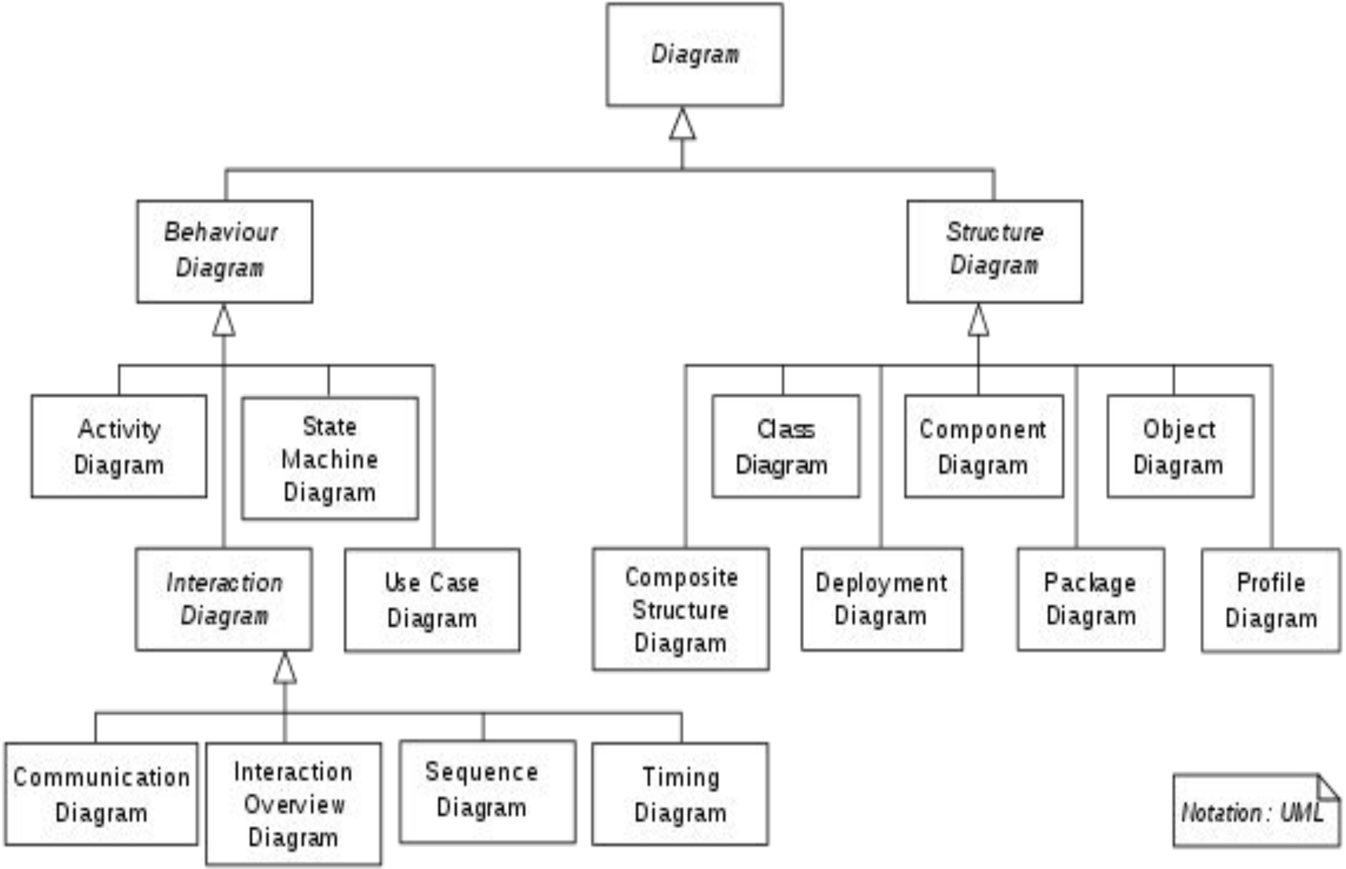
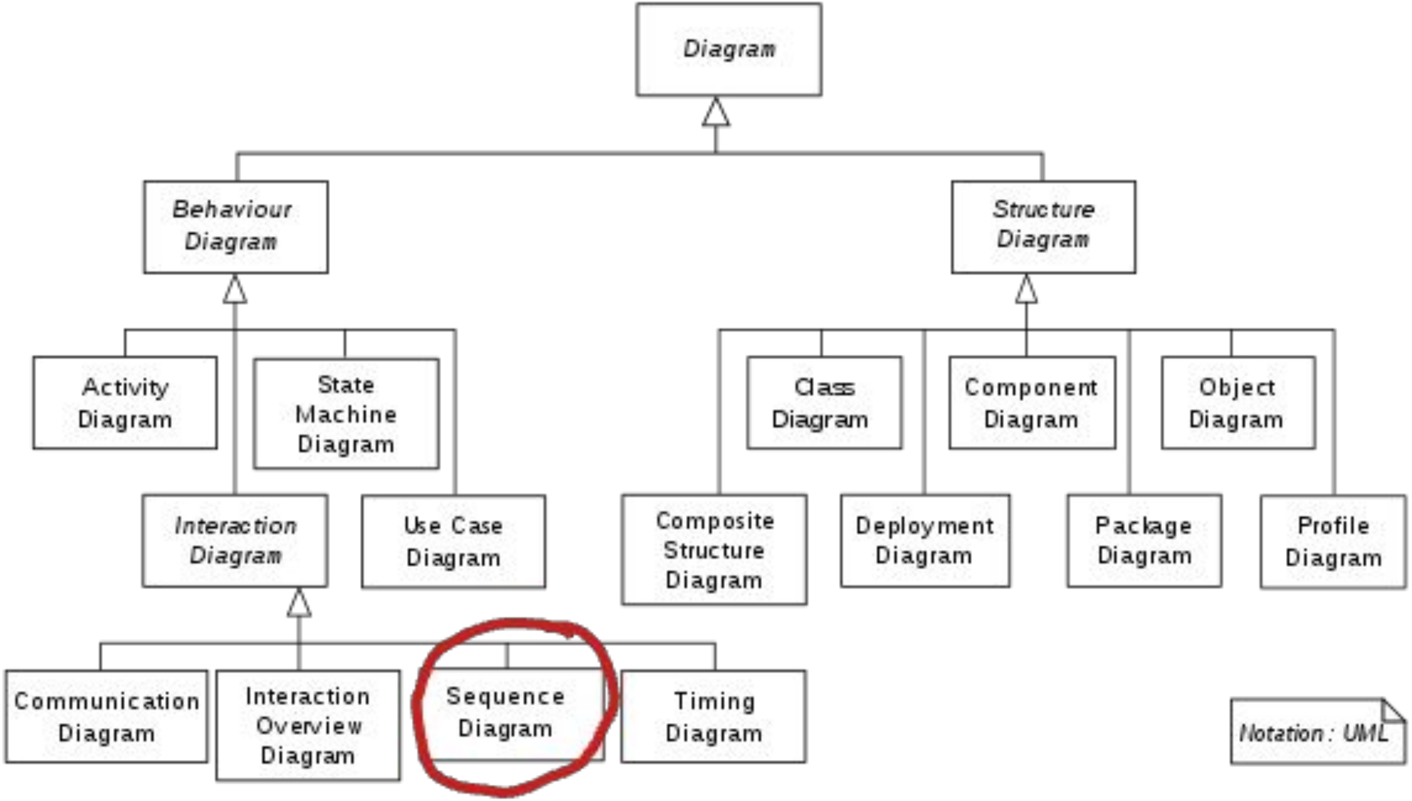


Diagram
Behaviour Diagram
Structure Diagram
Activity Diagram
State Machine Diagram
Interaction Diagram
Use Case Diagram
Communication Diagram
Interaction Overview Diagram
Sequence Diagram
Timing Diagram
Class Diagram
Component Diagram
Object Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Profile Diagram
Notation : UML

In today's class …



Standard Reference Link

In case of conflicts, we will use this for resolving them:

uml-diagrams.org

Interaction Diagrams

Are used to model the **dynamic** aspects of a software system

To visualize how the system runs.

An interaction diagram is often built from a use case and a class diagram [which we are yet to see].

□ The objective is to show how a **set of objects** accomplish the required interactions with an **actor**.

Interactions through Messages

- Interaction diagrams show how a set of actors and objects communicate with each other to perform:
  - **The steps of a use case,** or
  - The steps of some other piece of functionality.

- The set of steps, taken together, is called **an *interaction***.

- Interaction diagrams can show several different types of communication.
  - Method calls, messages send over the network
  - These are all referred to as *messages*.

Elements of Interaction Diagrams

- Instances of classes
  - Shown as boxes with the class and object identifier underlined

- Actors
  - Use the stick-person symbol as in use case diagrams

- Messages
  - Shown as arrows from actor to object, or from object to object

Before creating an interaction diagram

Ideally, we would have

Class Diagram

Use Case Diagram

Three main types of Interaction Diagrams

- Sequence
- Timing
- Communication

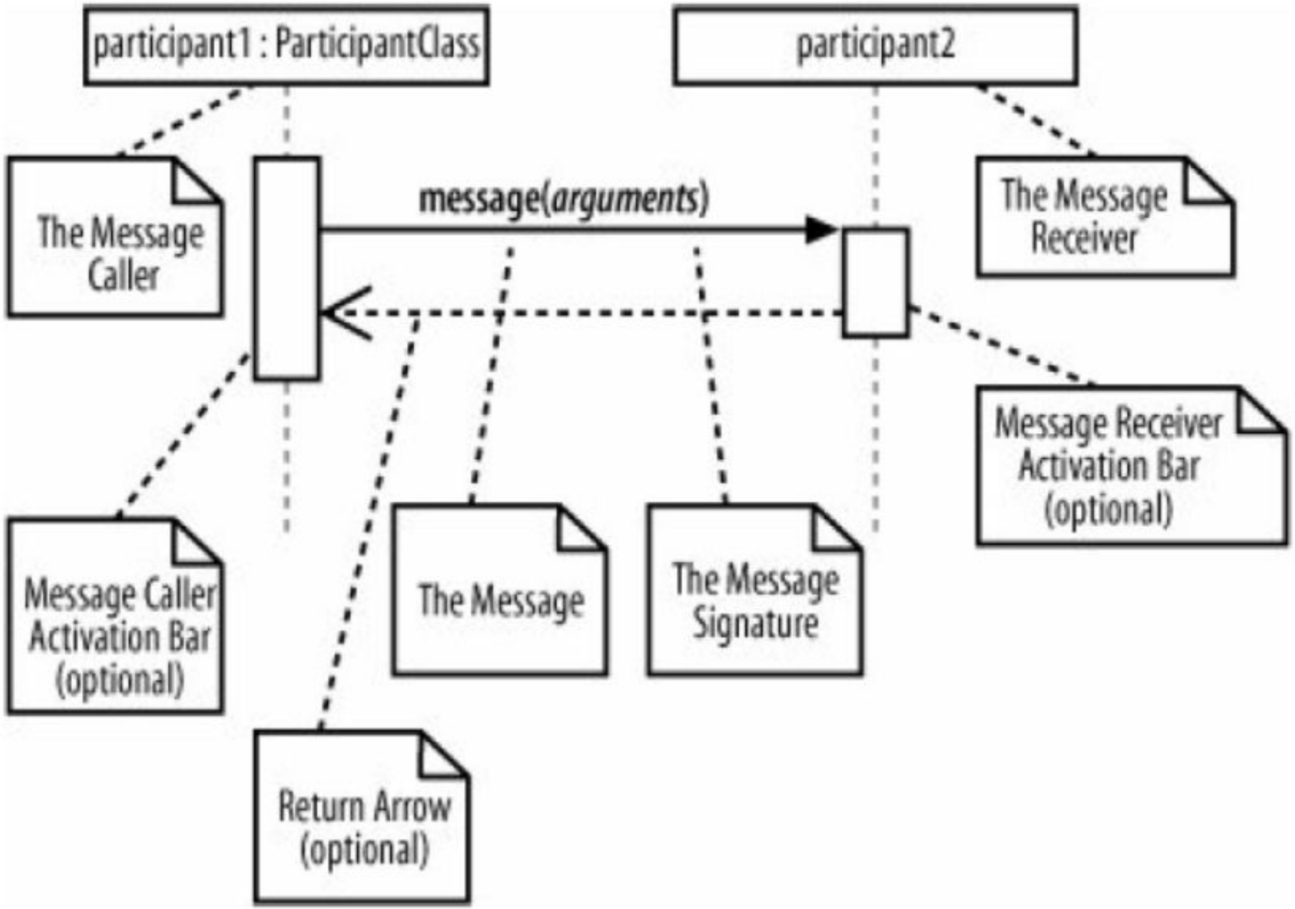
Sequence Diagram

Shows the sequence of messages exchanged by the set of objects performing a certain task

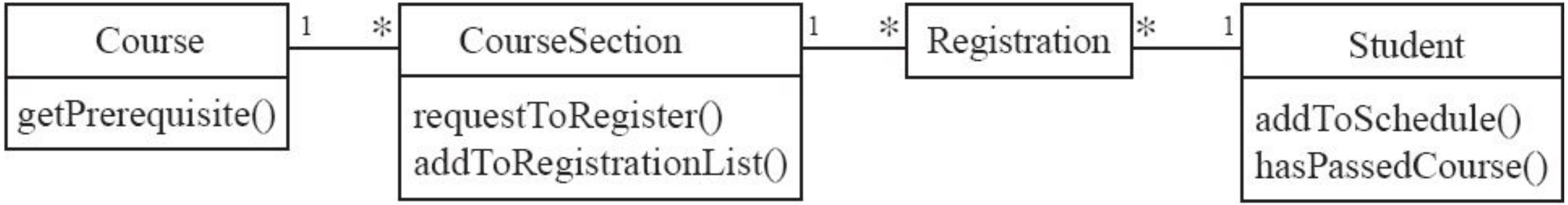
The objects are arranged horizontally across the diagram.
An actor that initiates the interaction is often shown on the left.
The vertical dimension represents time.

A vertical line, called a *lifeline*, is attached to each object or actor.
The lifeline becomes a broad box, called an *activation box*
during the *live activation* period.

A message is represented as an arrow between activation boxes of the sender and receiver.
A message is labelled and can have an argument list and a return value.

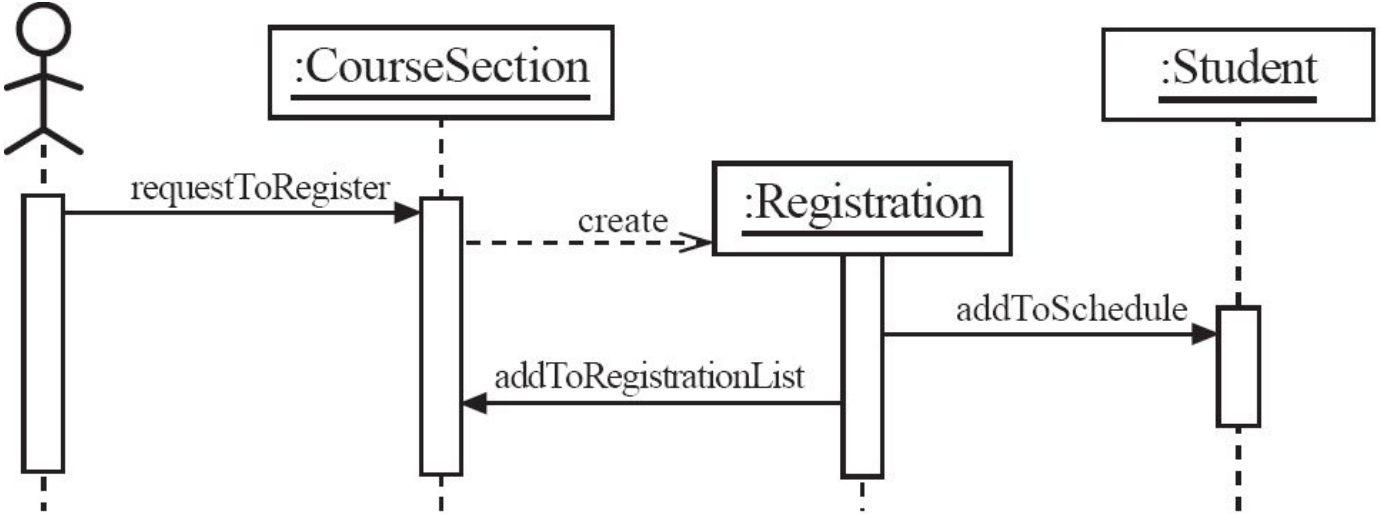

participant1 : ParticipantClass
participant2
The Message Caller
message(*arguments*)
The Message Receiver
Message Receiver Activation Bar (optional)
Message Caller Activation Bar (optional)
The Message
The Message Signature
Return Arrow (optional)

Assume you have the following class diagram

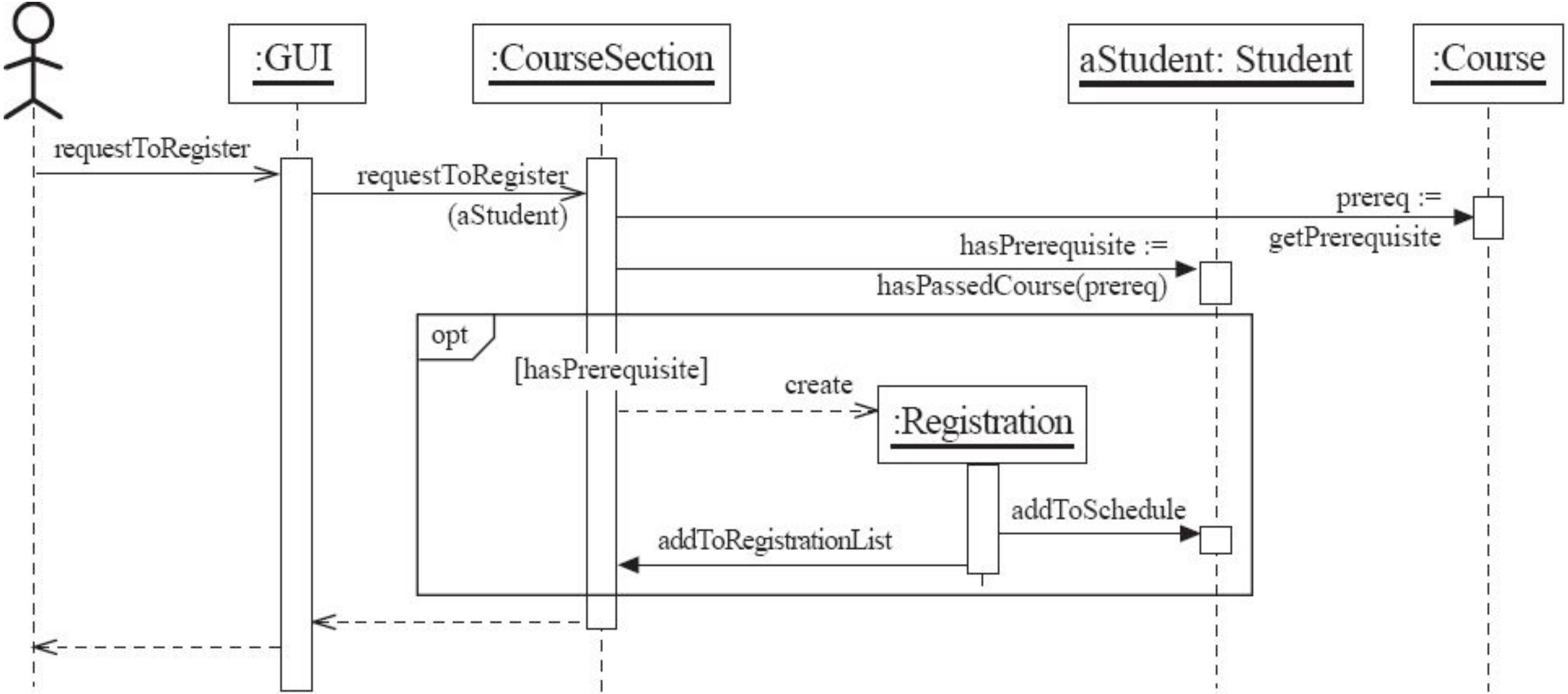


And you want to capture some of those functions

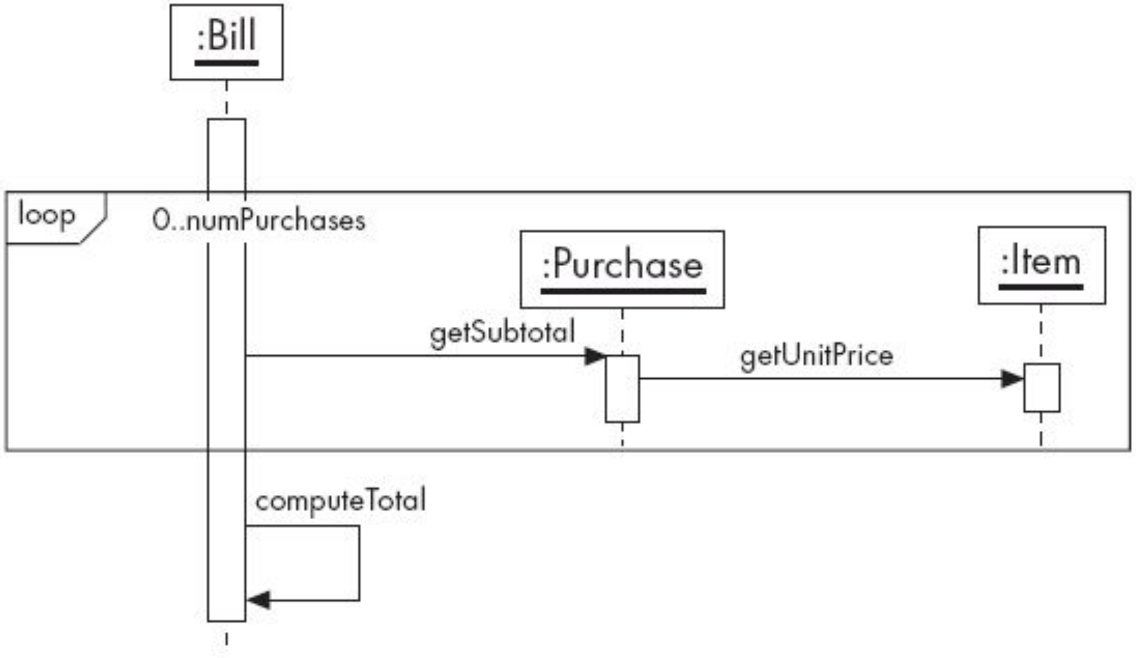
Somewhat “high level”



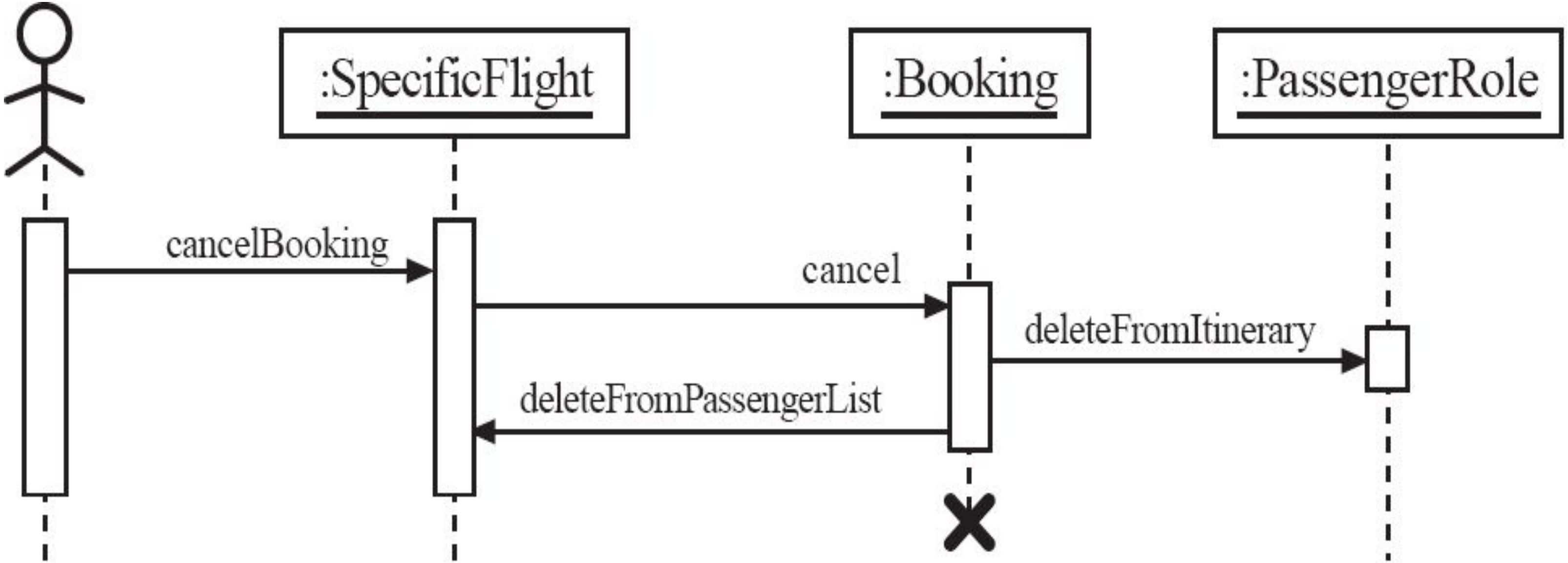
In a “detailed level”



Loops and Computations



When an object ‘dies’ that is, when its life ends …



Few Points

Deciding the level of details of the functionality

Deciding the level of details in the diagram

Remember the Standard Reference

What next?

Design - “the most important part” of SDLC has just started

We have Use Case Diagram and Sequence Diagram already

We have the User Interfaces too.

Class Diagram and a few more diagrams will be taught in the next week

In the Design document, include these diagrams

Start implementation / coding