Introduction of SSD

- A system sequence diagram is a visual or pictorial representation for a particular scenario of use case which shows the event that external actor generate, order of such event and external event.
- It illustrate input and output events related to the system and how such event are handle by the system.
- It depicts the processes involved and the sequence of messages exchanged between the processes needed to carry out the functionality.

SSD Specify following things

- External Actors
- Message or methods invoked by these actors
- If any return value associated with previous messages.
- Indication of any loops or iteration

SSD Specify following things

- The events which are generated during use case diagram are handle by SSD by managing external and internal system and their interactions.
- It can be drawn for particular scenario of use case or main success scenario of use case.
- The UML sequence diagram as a notation is used to visually represent system sequence diagram which illustrate actor interaction and the operation initiated by them.

Relationship Between Use case and SSD

- Use case is a collection of related success and failure scenarios where as SSD is a request and response events of that activity.
- Both diagram shows set of sequence of interaction between the system actor in a particular environment.
- Both use actor to show activities in define environment.
- For the particular use case SSD shows what events are generated by the external entity.
- SSD use UML sequence diagram notation to visually represent the main success scenario of the use case.

Use Case Diagram	System Sequence Diagram
• A use case diagram is a way to summarize details of a system and the users within that system.	• **System Sequence Diagram** (SSD) shows process interactions arranged in time sequence in the field of software Engineering.
• Use case diagram does not care about responses.	• SSD request and response events with different notations.
• It is drawn in Inception phase of UP Process.	• It is drawn in Elaboration phase of UP Process.
• It Shows the relationship of Use cases with defined notation.	• It shows sequence of events.

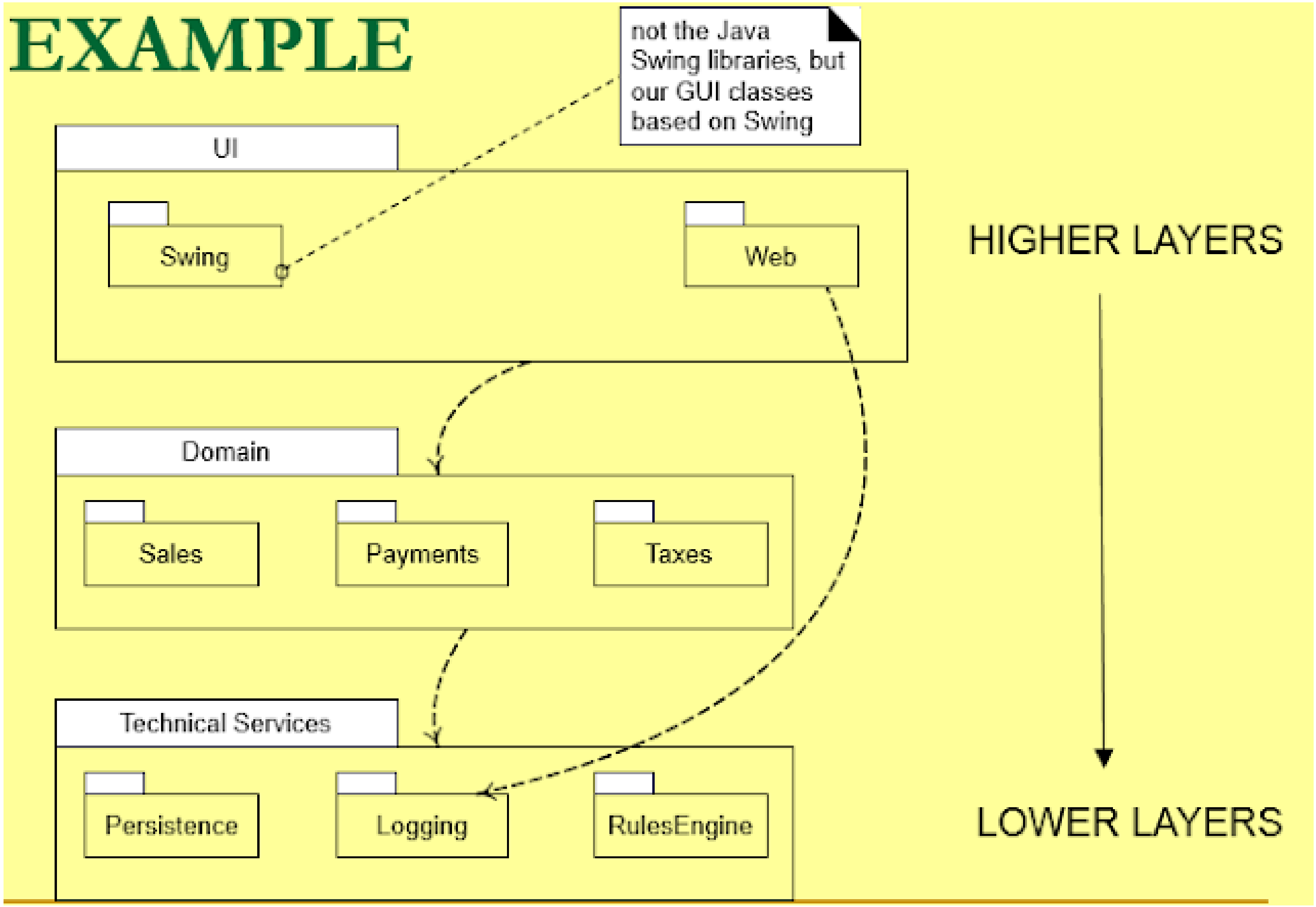
Logical Architecture and Layer

- The **logical architecture** is the large-scale organization of the software classes into packages (or namespaces), subsystems, and layers.
- It's called the *logical* architecture because there's no decision about how these elements are deployed across different operating system processes or across physical computers in a network (these latter decisions are part of the **deployment architecture**).
- A **layer** is a very coarse-grained grouping of classes, packages, or subsystems that has cohesiveresponsibility for a major aspect of the system. Also, layers are organized such that "higher"layers (such as the UI layer) call upon services of "lower" layers, but not normally vice versa.

Layers in OO System include

- **User Interface** or UI
- **Application Logic and Domain Objects/Business Layer or Business Logic Layer** software objects representing domain concepts(For example, a software class *Sale*) that fulfill application requirements, such as calculating a sale total.
- **Technical Services/Data Access Layer or Data Layer** general purpose objects and subsystems that provide supporting technical services, such as interfacing with a database or error logging. These services are usually application-independent and reusable across several systems.

Logical Architecture and Layer



Types of Layered Architecture

- This layer only calls upon the services of the layer directly below it.
- This design is common in network protocol stacks,

- .In which a higher layer calls upon several lower layers.
- This Design used in information systems,
- For example, the UI layer may call upon its directly subordinate application logic layer, and
- also upon elements of a lower technical service layer, for logging and so forth.

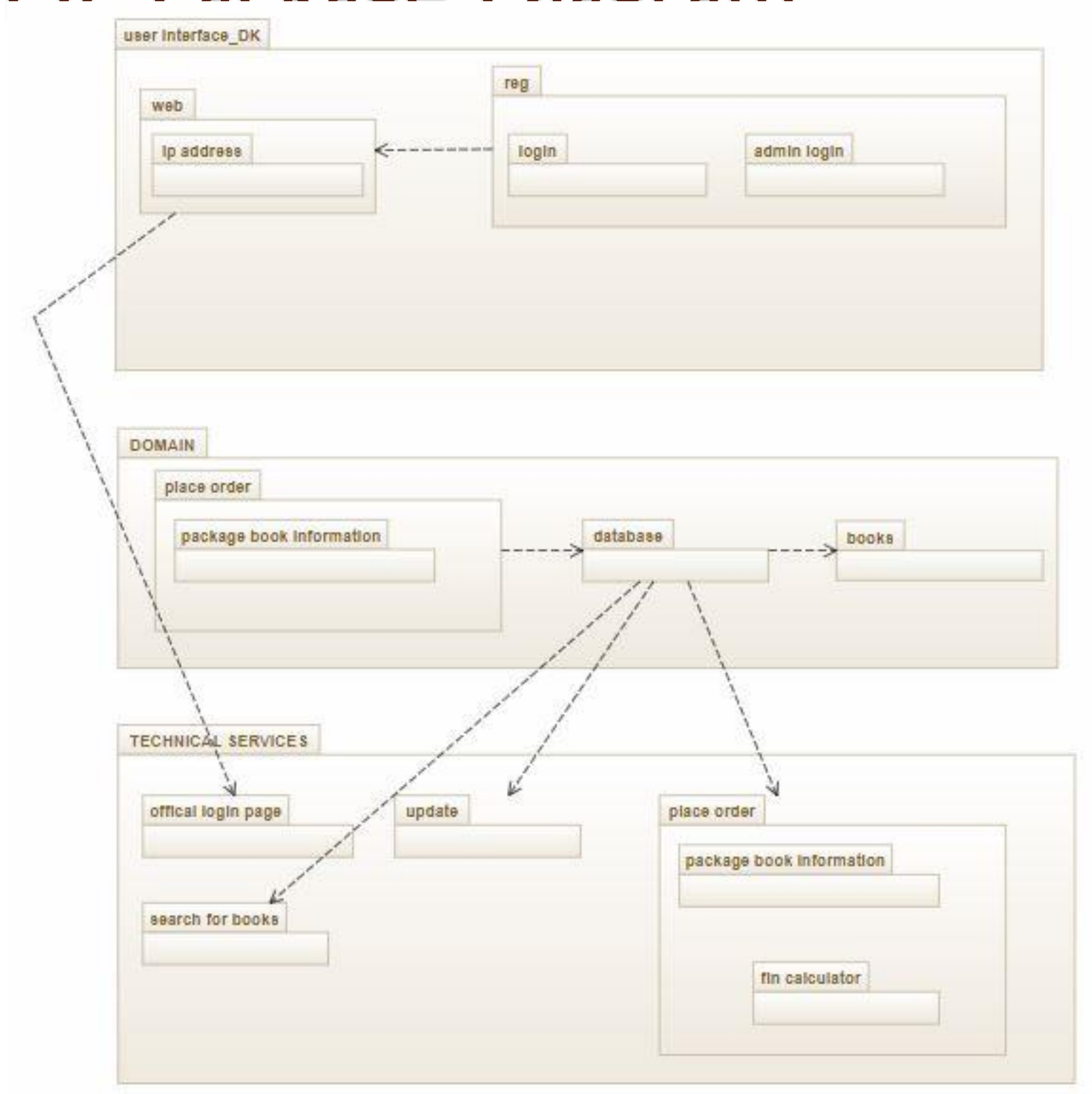
UML Package Diagram

- Package Diagram is the kind of structural diagram which shows the structure, arrangement and organization of the designed system, subsystem, group of classes that form the system and dependencies between the packages.
- Package diagram can show both the structure and dependencies between the subsystem or modulus and subsystem or modulus and also different views of the system.

UML Package Diagram

- A UML Package can group anything like classes, subsystem, other package use cases etc.
- The UML Package diagram is represented by tabbed folder and package name may be placed on the tabbed if the package show inner member or in the main folder.
- **Dependency** between package in different layer might exist which shows the one package is depends on another package uses the function or services provided by the other packages.

UML Package Diagram



Designing with layers

- The main purpose of layered architecture is to organize the large scale logical structure of a system into different layer where lower level layers are used for general service and higher level layers are used for application specific.
- The purpose and number of layers may varies according to application logic and application domain.

Common layers of Information System Logical Architecture

- Presentation Layer
- Application Layer
- Domain Layer
- Business Infrastructure Layer
- Technical Service Layer
- Foundation Layer

Presentation Layer

- It provides general UI interface for the user input, creating the window and Widgets, capturing Mouse and Keyboard Events.
- It Trigger the Event for the system which is generated by the user input and shows the responses provided by the system as per request.

Application Layer

- It handles the request of presentation layer , session states and page or window transition.
- It provides the way of transforming data and way of presenting data.
- It decides when to send data to domain layer?

Domain Layer

- It provide the service based on domain rules.
- It provides the general services and operations that the system have to provide.

Business Infrastructure Layer

- It is responsible for performing business domain which provide extra service to the domain layer.
- Some business function that can be done by business infrastructure layer are currency converter, date calculation etc.

Technical Service Layer

- It perform high level technical service and framework like security services.
- If business need any operation it my provide database operation such as mysql commands and lower level of mysql operation.

Foundation Layer

- It provide basic level operation such as file and folder operation for information system.
- It checks the mysql request receive from mysql operation such as insert or select options.

Class Diagram

- Class Diagram is a static Diagram.
- It represents the static view of an application.
- Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class Diagram

- Class diagram describes the attributes and operations of a class and also the constraints imposed on the system.
- The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.
- Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Purpose of Class Diagram

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering.

How to Draw a Class Diagram?

- It is very important to learn the drawing procedure of class diagram.
- Class diagrams have a lot of properties to consider while drawing but here the diagram will be considered from a top level view.
- Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. A collection of class diagrams represent the whole system.

Points to be remember while drawing Class Diagram

- The name of the class diagram should be meaningful to describe the aspect of the system.
- Each element and their relationships should be identified in advance.
- Responsibility (attributes and methods) of each class should be clearly identified
- For each class, minimum number of properties should be specified, as unnecessary properties will make the diagram complicated.

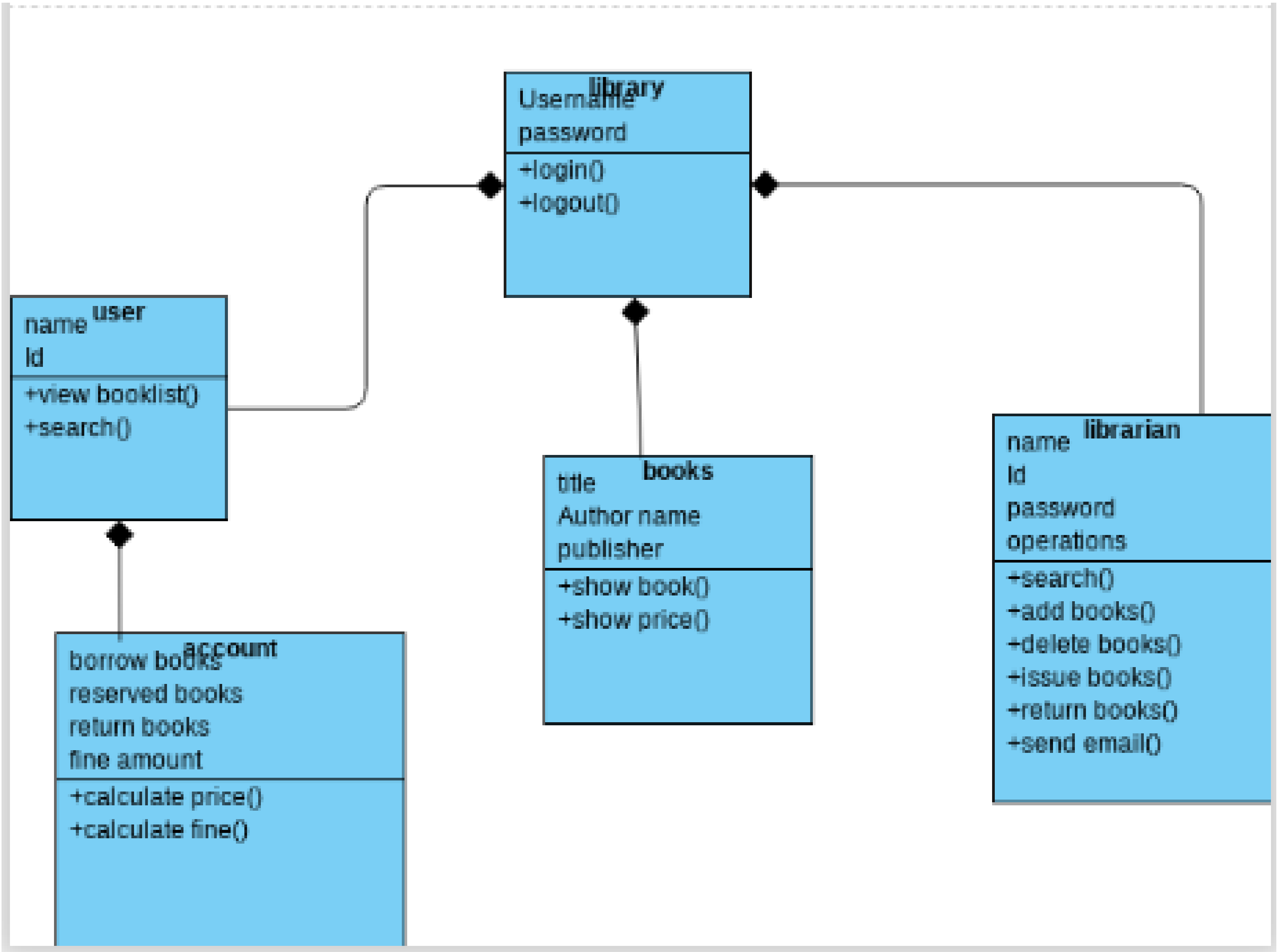
Points to be remember while drawing Class Diagram

- Use notes whenever required to describe some aspect of the diagram. At the end of the drawing it should be understandable to the developer/coder.
- Finally, before making the final version, the diagram should be drawn on plain paper and reworked as many times as possible to make it correct.

Where to Use Class Diagrams?

- Describing the static view of the system.
- Showing the collaboration among the elements of the static view.
- Describing the functionalities performed by the system.
- Construction of software applications using object oriented languages.

Class Diagram Library Management System



Interaction Diagram

- Interaction diagram is used to describe some type of interactions among the different elements in the model. So this interaction is a part of dynamic behaviour of the system.
- This interactive behaviour is represented in UML by two diagrams known as Sequence diagram and Collaboration diagram.
- The basic purposes of both the diagrams are similar.

Purpose of Interaction Diagram

- To capture dynamic behaviour of a system.
- To describe the message flow in the system.
- To describe structural organization of the objects.
- To describe interaction among objects.

Interaction Diagram

- Dynamic aspect can be defined as the snap shot of the running system at a particular moment.
- The sequence diagram captures the time sequence of message flow from one object to another and the collaboration diagram describes the organization of objects in a system taking part in the message flow.

How to draw Interaction Diagram

- Dynamic aspect can be defined as the snap shot of the running system at a particular moment.
- The sequence diagram captures the time sequence of message flow from one object to another and the collaboration diagram describes the organization of objects in a system taking part in the message flow.

How to draw Interaction Diagram

- Objects taking part in the interaction.
- Message flows among the objects.
- The sequence in which the messages are flowing.
- Object organization.

Uses of Interaction Diagram

- To model flow of control by time sequence.
- To model flow of control by structural organizations.
- For forward engineering.
- For reverse engineering.

The Collaboration Diagram

- The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram.
- Now to choose between these two diagrams the main emphasis is given on the type of requirement. If the time sequence is important then sequence diagram is used and if organization is required then collaboration diagram is used.

System Sequence Diagram	Collaboration Diagram
• SSD is used to visualize the sequence of calls in a system that is used to perform a specific functionality.	• CD is used to visualize organization of the objects and their interaction.
• SSD are used to represent the sequence of messages that are flowing from one object to another.	• CD are used to represent the structural organization of the system and the messages that are sent and received.
• The sequence diagram is used when time sequence is main focus.	• The collaboration diagram is used when object organization is main focus.
• The sequence diagrams are better suited of analysis activities.	• The collaboration diagrams are better suited for depicting simpler interactions of the smaller number of objects.