

| **Title: Study of Umbrello Unified Modelling Language tool Or Lucidchart: flowcharts and diagram drawing tool** |
| --- |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Aim:** To learn and understand the way of creating various UML diagrams for requirement analysis

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**CO:** Analyse the software requirements and Model the defined problem with the help of UML diagram.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. Roger Pressman, “Software Engineering”, sixth edition, Tata McGraw Hill.
2. System Analysis & Design by Satzinger, Jackson and Burd, Cengage Learning, 2007
3. System Analysis and Design Methods by Jeffery l. Whitten, Lonnie D Bentley,McGraw Hill, 7th edition.
4. System Analysis and Design by Alan Dennis, Barbara H. Wixom, Roberta M. Roth,Wiley India 4th edition
5. <http://en.wikipedia.org/wiki/Software_requirements_specification>
6. <http://en.wikipedia.org/wiki/Use_case>

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Pre Lab/ Prior Concepts:**

In [software](http://en.wikipedia.org/wiki/Software_engineering) and [systems engineering](http://en.wikipedia.org/wiki/Systems_engineering), a **use case** is a list of steps, typically defining interactions between a role (known in [Unified Modeling Language](http://en.wikipedia.org/wiki/Unified_Modeling_Language) (UML) as an "[actor](http://en.wikipedia.org/wiki/Actor_(UML))") and a system, to achieve a goal. The actor can be a human or an external system.

In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or [stakeholder](http://en.wikipedia.org/wiki/Project_stakeholder) goals. The detailed requirements may then be captured in [Systems Modeling Language](http://en.wikipedia.org/wiki/Systems_Modeling_Language) (SysML) or as contractual statements.

As an important requirement technique, use cases have been widely used in modern software engineering over the last two decades. Use case driven development is a key characteristic of process models and frameworks like [Unified Process](http://en.wikipedia.org/wiki/Unified_Process) (UP), [Rational Unified Process](http://en.wikipedia.org/wiki/Rational_Unified_Process) (RUP), [Oracle Unified Method](http://en.wikipedia.org/wiki/Oracle_Unified_Method) (OUM), etc. With its iterative and evolutionary nature, use case is also a good fit for [agile development](http://en.wikipedia.org/wiki/Agile_software_development).

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence of what happens first, what happens next.

Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

In UML, class diagrams are one of six types of structural diagram. Class diagrams are fundamental to the object modelling process and model the static structure of a system. Depending on the complexity of a system, you can use a single class diagram to model an entire system, or you can use several class diagrams to model the components of a system.

Class diagrams are the blueprints of your system or subsystem. You can use class diagrams to model the objects that make up the system, to display the relationships between the objects, and to describe what those objects do and the services that they provide.

In its basic form, an activity diagram is a simple and intuitive illustration of what happens in a workflow, what activities can be done in parallel, and whether there are alternative paths through the workflow. Activity diagrams as defined in the Unified Modeling Language are derived from various techniques to visually illustrate workflows. Activity diagrams are used to visualize the workflow of a business use case. A complete workflow description will have a basic flow, and one or several alternative flows. This workflow has a structure that we can define textually, using informal if, if-then-else, or does-until statements of various kinds. For a simple workflow with a simple structure such textual definitions may be quite sufficient, but in the case of more complex structures, activity diagrams help to clarify and make more apparent what the workflow is. Historically, activity diagramming techniques have mostly been used in the business process modeling domain, but this article will also briefly discuss how you can use it in the system modeling domain.

**Requirement Modeling:**

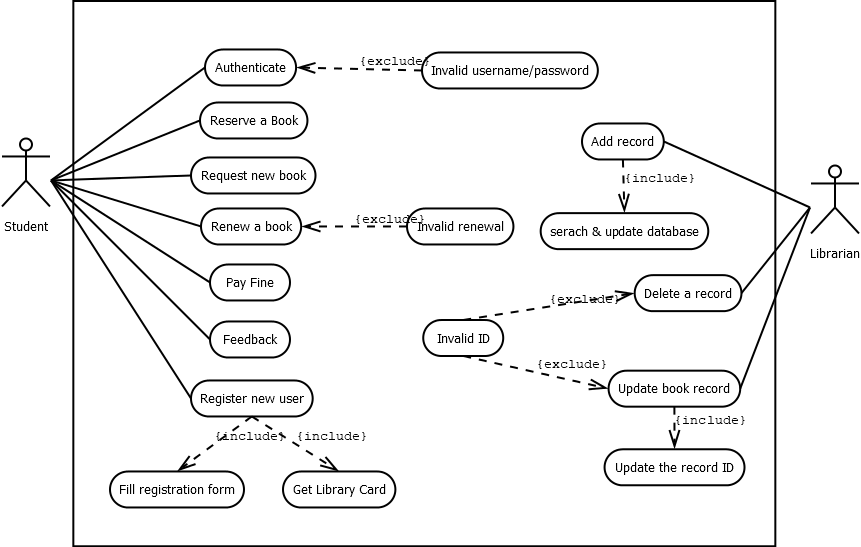
*Significance of every diagram is to be written*

1. **Use Case**

 **Purpose:** Captures the functional requirements of a system from the perspective of its users (actors).

 **Significance:**

* Defines the system's boundaries and interactions with its environment.
* Identifies the primary functions the system must perform.
* Provides a high-level overview of the system's functionality.
* Serves as a basis for developing other diagrams and artifacts.

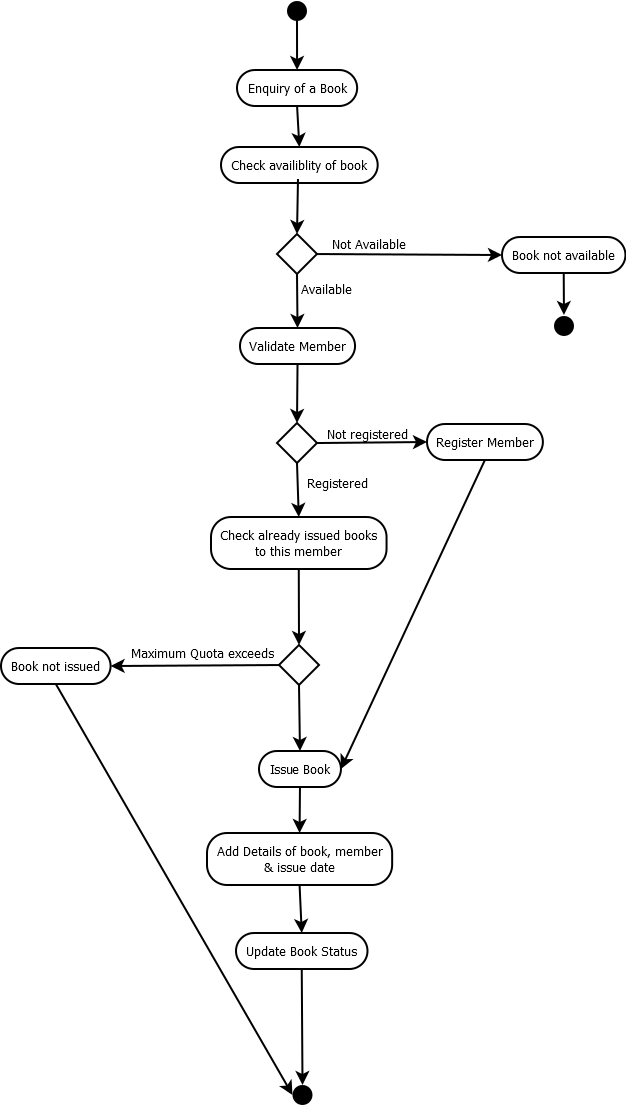


1. **Activity Diagram**

 **Purpose:** Models the workflow and sequence of activities within a use case.

 **Significance:**

* Visualizes the flow of control and decision points.
* Helps in understanding the system's internal processes.
* Identifies potential bottlenecks and inefficiencies.
* Supports the development of detailed design specifications.

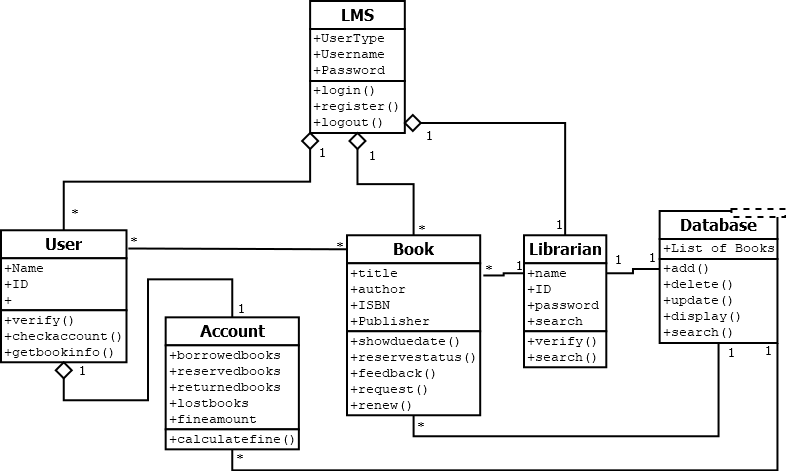


1. **Class Diagram**

 **Purpose:** Represents the static structure of a system by showing its classes, attributes, and relationships.

 **Significance:**

* Defines the objects and their properties in the system.
* Identifies the relationships between classes (e.g., inheritance, association, aggregation).
* Serves as a blueprint for the system's implementation.
* Provides a foundation for designing the system's data model.

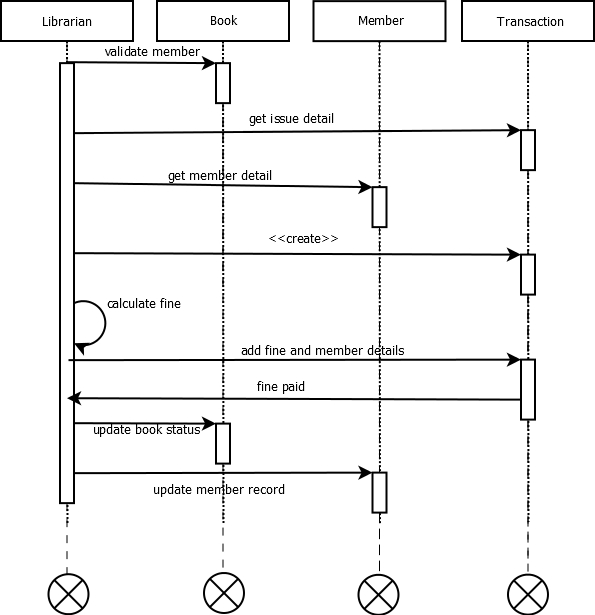


1. **Sequence**

 **Purpose:** Models the interactions between objects in a system over time.

 **Significance:**

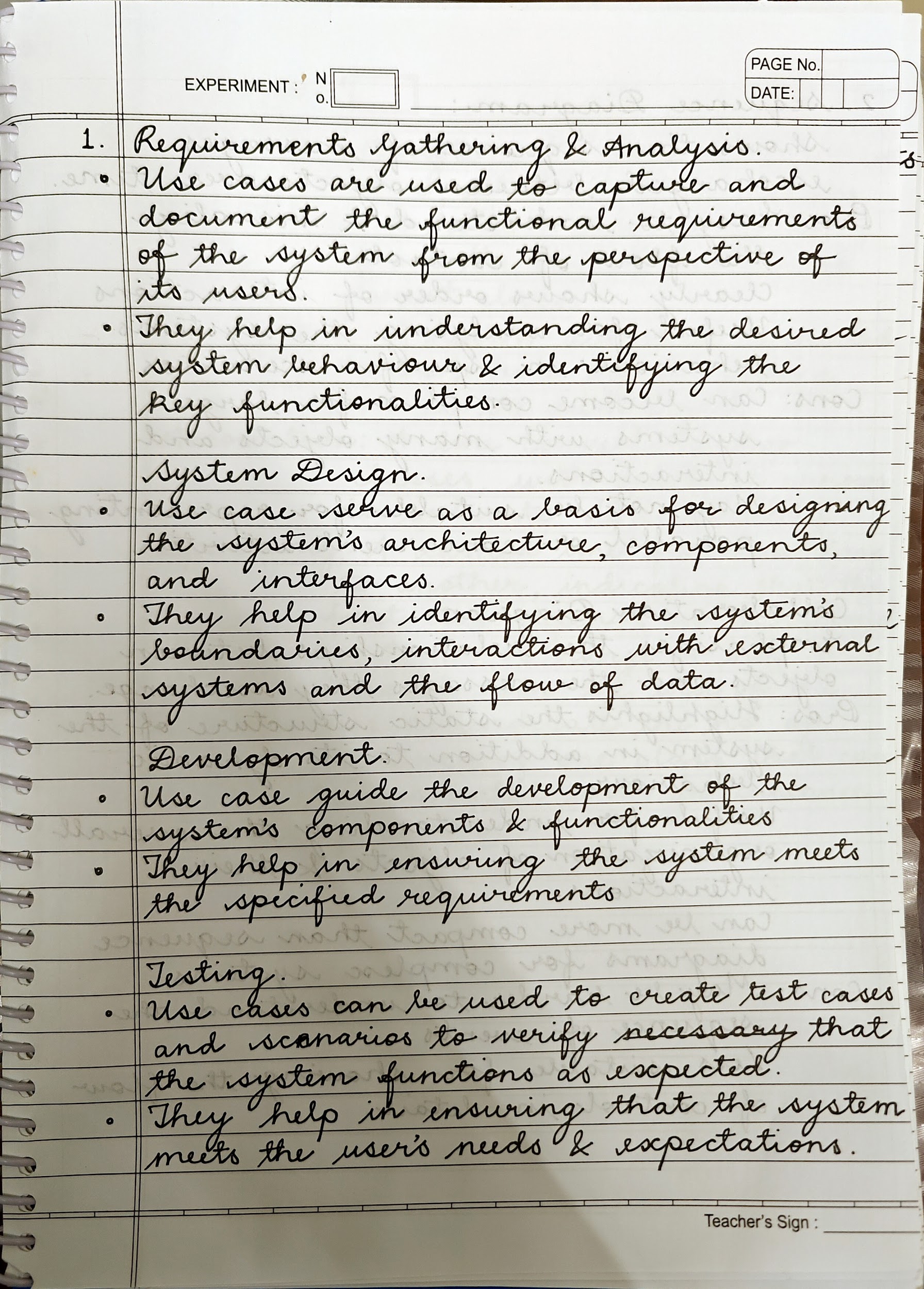
* Shows the sequence of messages exchanged between objects.
* Helps in understanding the system's dynamic behavior.
* Identifies potential performance issues and synchronization problems.
* Supports the development of detailed design specifications.



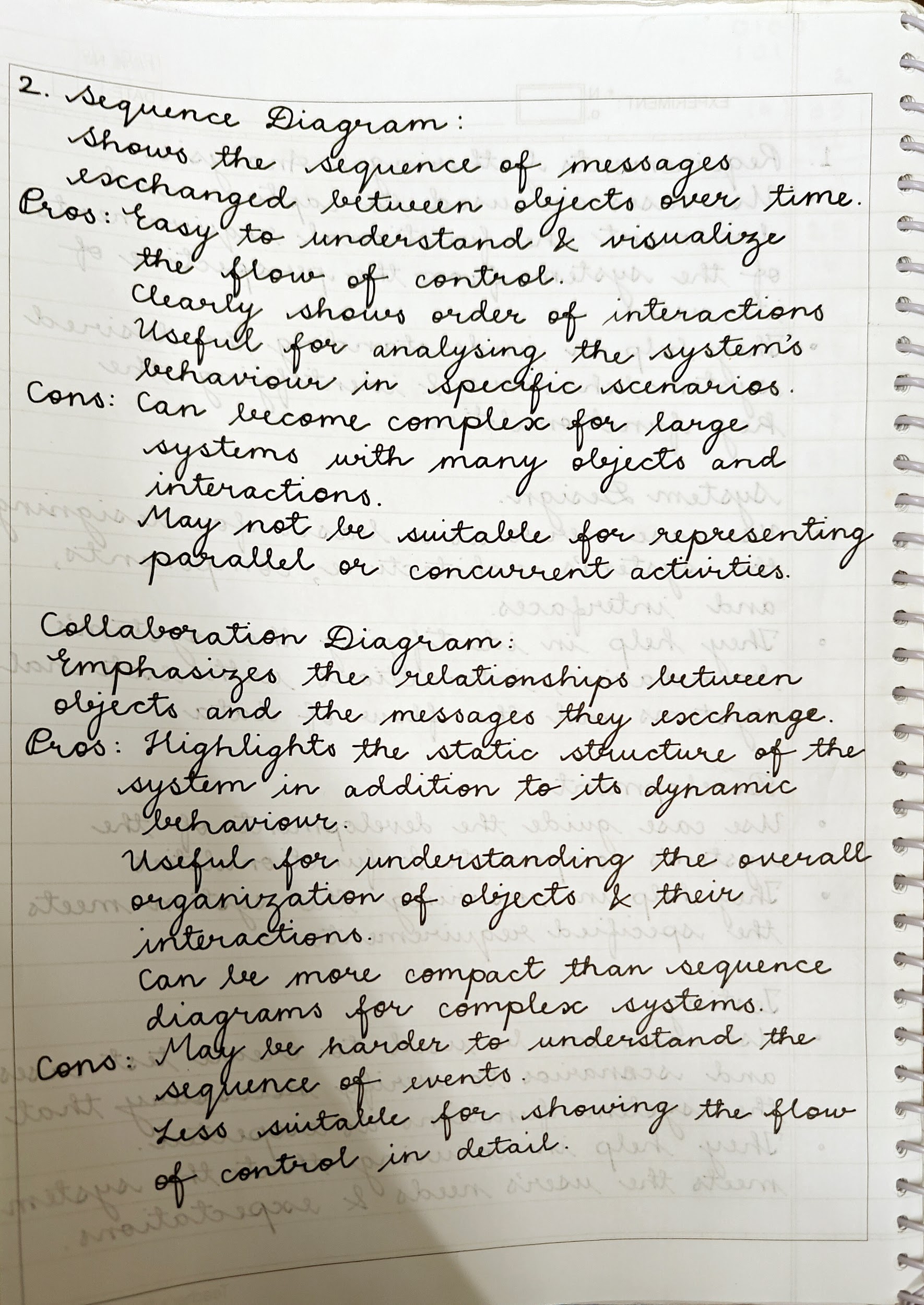
**Conclusion:** We learned the significance of various UML diagrams and how they function.

**Post Lab Descriptive Questions:**

1. Where do use cases fit in the software development life cycle?

Ans:  


1. Compare sequence diagram with collaboration diagram. Explain pros and cons of each.

Ans:  


1. List different notations used in Class diagram with example

Ans:

