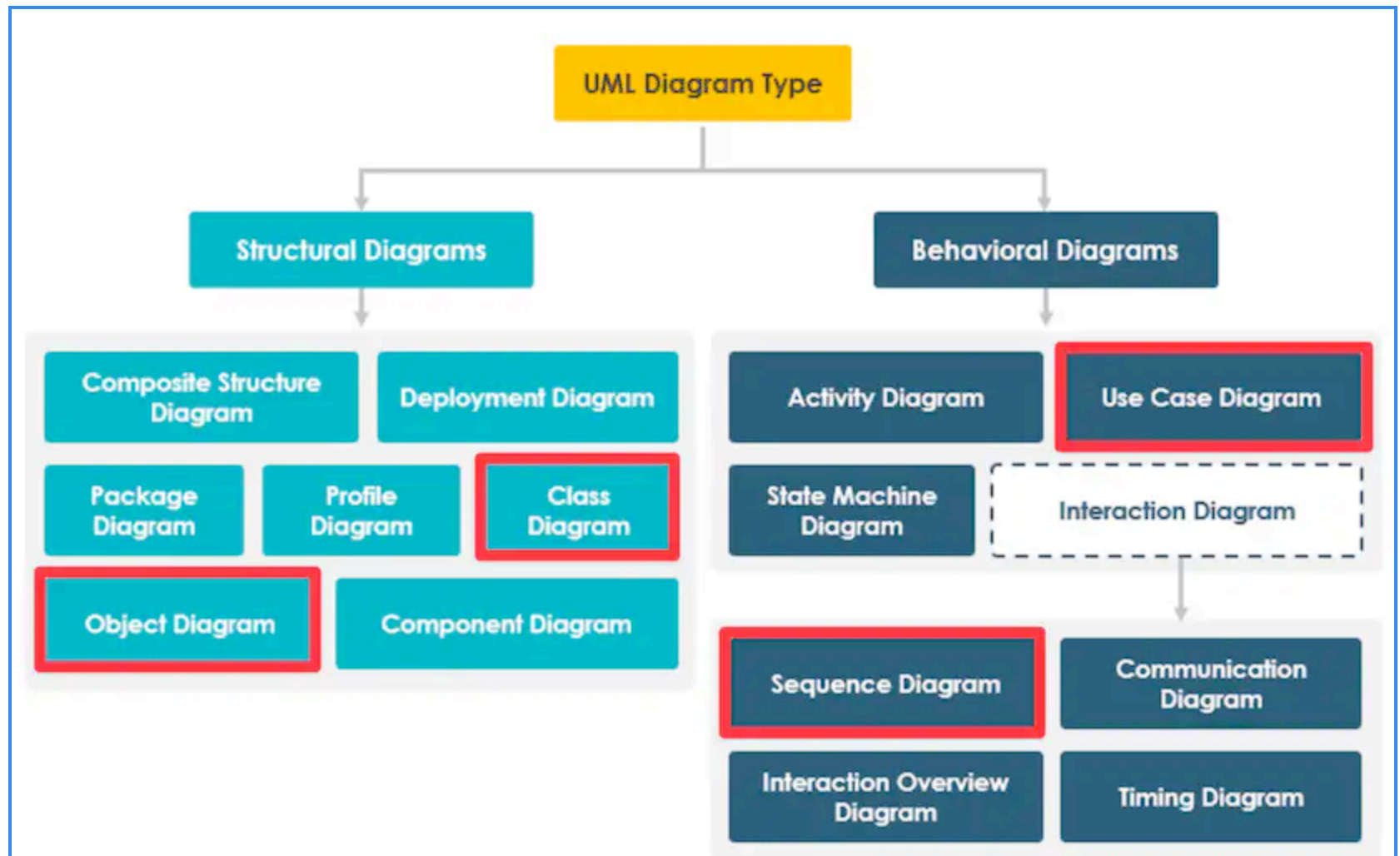


UML Diagrams

Behavioral Diagrams

- So far, we have discussed "Structural Diagrams" in UML.
 - Class Diagram & Object Diagram and their relationships.
- Now, we discuss "Behavioral Diagrams".
 - Use Case Diagram & Sequence Diagram



Use case Diagram

A Use Case diagram shows who can do what with a system, capturing functional requirements via actors, use cases, and their relationships.

For example, in a library system.

- We need to ask questions: "Who will use this system?" and "What do they want to do?"
- Students want to find and borrow books, and Librarians want to manage the books.

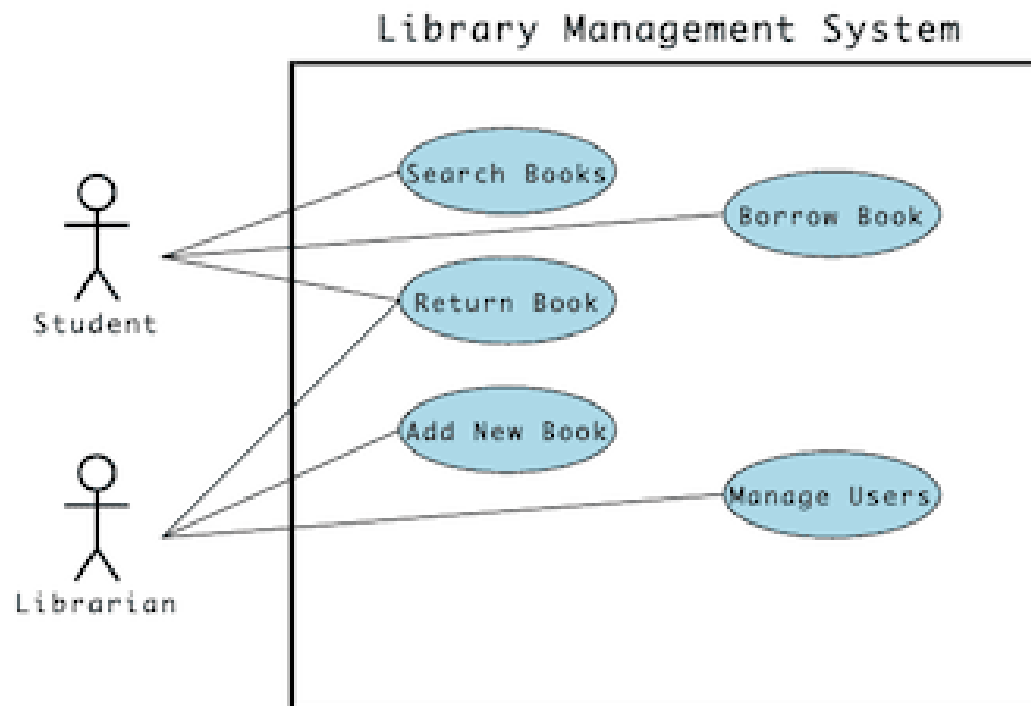
We can rewrite this question into the "Actor-Goal" Format.

- "As a [actor], I want to [use case] so that [benefit]"
- "As a student, I want to **search for books** so that I can find relevant materials for my research."

We can use the same "Actor-Goal" format for the librarian.

- "As a librarian, I want to add new books so that I can refresh my database, so that students can access it."
- "As a Librarian, I want to manage users so that I can keep track of students' database records."

Using the format, we can make a Use case diagram.



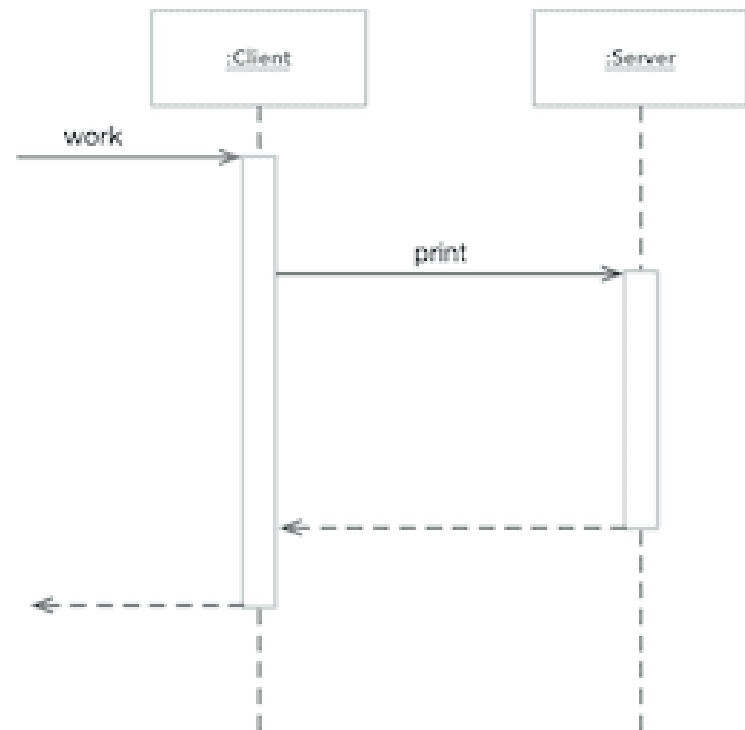
Why use case diagrams?

- Communication between developers and stakeholders
- Aid for gathering requirements
- Tool for defining system scope
- Guide for testing user scenarios

Sequence Diagram

- UML sequence diagrams show method calls between objects
- Example:
 - Client object calls work()
 - work() calls Server's print()

```
class Client(object):  
    def work(): Server().print();  
  
class Server(object):  
    def open(): ...  
  
c = Client()  
c.work();
```



- A sequence diagram is about the interaction among objects.
 - It uses the object diagram convention (object type underlined).
- A solid line is used for invocation, and a dotted line is used for the return.

Why sequence diagrams?

- Shows what objects use which methods and when
- Prevents design issues (e.g., circular dependencies)
- Improves team communication